AFRL-ML-WP-TR-1998-4095

NONDESTRUCTIVE EVALUATION FOR DATA FUSION

James M. Nelson
Richard H. Bossi
Christy A. Lancaster

Boeing Information, Space & Defense Systems
PO Box 3999
Seattle WA 98124

MAY 1998

FINAL REPORT FOR PERIOD SEPTEMBER 1995 – APRIL 1998

MATERIALS & MANUFACTURING DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7734

NOTICE

WHEN GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA ARE USED FOR ANY PURPOSE OTHER THAN IN CONNECTION WITH A DEFINITELY GOVERNMENT-RELATED PROCUREMENT, THE UNITED STATES GOVERNMENT INCURS NO RESPONSIBILITY OR ANY OBLIGATION WHATSOEVER. THE FACT THAT THE GOVERNMENT MAY HAVE FORMULATED OR IN ANY WAY SUPPLIED THE SAID DRAWINGS, SPECIFICATIONS, OR OTHER DATA, IS NOT TO BE REGARDED BY IMPLICATION OR OTHERWISE IN ANY MANNER CONSTRUED, AS LICENSING THE HOLDER OR ANY OTHER PERSON OR CORPORATION, OR AS CONVEYING ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY IN ANY WAY BE RELATED THERETO.

THIS REPORT IS RELEASABLE TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS). AT NTIS, IT WILL BE AVAILABLE TO THE GENERAL PUBLIC, INCLUDING FOREIGN NATIONS.

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.

Laura L. Mann

LAURA L. MANN, Project Engineer
Nondestructive Evaluations Branch
Metals, Ceramics & NDE Division

Tobey M. Cordell

TOBEY M. CORDELL, Chief
Nondestructive Evaluations Branch
Metals, Ceramics & NDE Division

Norman M. Geyer

NORMAN M. GEYER, Asst Chief
Metals, Ceramics & NDE Division
Materials & Manufacturing Directorate

IF YOUR ADDRESS HAS CHANGED, IF YOU WISH TO BE REMOVED FROM OUR MAILING LIST, OR IF THE ADDRESSEE IS NO LONGER EMPLOYED BY YOUR ORGANIZATION, PLEASE NOTIFY, AFRL/MLLP, WRIGHT-PATTERSON AFB OH 45433-7817 TO HELP US MAINTAIN A CURRENT MAILING LIST.

COPIES OF THIS REPORT SHOULD NOT BE RETURNED UNLESS RETURN IS REQUIRED BY SECURITY CONSIDERATIONS, CONTRACTUAL OBLIGATIONS, OR NOTICE ON A SPECIFIC DOCUMENT.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
May 1998

3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE
NONDESTRUCTIVE EVALUATION FOR DATA FUSION

5. FUNDING NUMBERS
C - F33615-95-C-5234
PE - 63112F
PR - 3153
TA - 00
WU - 16

6. AUTHOR(S)
James M. Nelson, Richard H. Bossi, Christy A. Lancaster

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Boeing Information, Space, & Defense Systems
P. O. Box 3999
Seattle, WA 98124

8. PERFORMING ORGANIZATION REPORT NUMBER
D658-10844-1

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Materials & Manufacturing Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson Air Force Base, OH 45433-7734
POC: Laura Mann, AFRL/MLLP, 937-255-9798

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-ML-WP-TR-1998-4095

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release: distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

.NDE data fusion methodologies and algorithms were refined and implemented under this programand have been packaged and delivered to the Air Force in a visually programmable, object oriented application builder software package entitled INDERS 3. An important functional element of this package is the ability to integrate part geometry models with associated NDT data.

Three case studies were selected and conducted during the development of the package. The first case study applies to the development of a low cost manufacturing process for the Joint Strike Fighter composite wingbox. The second case study examined the effect on radar performance of maintenance and repair of the E-3 AWACS radome. The third case study applied to evaluation of B1-B weapons bay doorsfor in service damage. The package was implemented on a purchased Silicon Graphics workstation which was used for the case studies and was delivered to the Air Force at the conclusion of the contract. The results of the effort were presented to both industry and Air Force personnel in a series of meetings, demonstrations, and software installations.

14. SUBJECT TERMS
NDE, Nondestructive Evaluation, Data Fusion, Ultrasonics, Visualization, Workstation

15. NUMBER OF PAGES
115

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

CONTENTS

FIGURES

FOREWORD

This report represents the work accomplished by the Phantom Works Division of the Boeing Information, Space, Defense, and Systems Group from 15 September 1995 to 30 May 1998 on development of Nondestructive Evaluation (NDE) data fusion methods. The work was performed under the technical direction of James Nelson and supervision of John Shrader of the Boeing Information, Space, Defense, and Systems Group, Phantom Works Division.

Technical effort was supported within Boeing by Richard H. Bossi, Christy Lancaster, Loren Milliman, and Ray Rempt. We also wish to express our appreciation to Dave Argyle and James Youngberg (Perceptics Inc.) for their technical assistance in performing this program.
This work is sponsored by the Air Force Research Laboratory, NDE Branch under contract F33615-95-C-5234. Laura Mann is the AFRL program manager.

EXECUTIVE SUMMARY

NDE data fusion can be defined as the process of reducing large quantities of NDT and related data into a unified model which can be conveniently evaluated by an end user. NDE data fusion has been identified as an important aerospace technology with demonstrated cost benefit for very high value asset evaluation. The NDE technical community believes that this technology has potential for reducing ambiguity and improving decision making processes in a wider context. This program has sought to implement and package NDE data fusion tools to make the technology more widely available to the aerospace community.

The primary objectives of implementing NDE data fusion tools and processes is to reduce time for non–experts to understand complex NDT data and to reduce time for experts to assemble NDE data fusion software applications. NDE data fusion methodologies and algorithms based on earlier work have been refined and implemented under this program. These have been packaged and delivered to the Air Force in a visually programmable, object oriented application builder software package entitled INDERS 3. An important functional element of this package is the ability to integrate part geometry models with associated NDT data.

Visual programming and data object visualization functionality are important functionalities for general use by the NDE community. The implementation of INDERS 3 under this program provides a model for expected natural evolution from the image processing paradigm to the object visualization paradigm. In particular, the drive to more affordable manufacturing is leading to complex, monolithic structures and graded fitness–for–service criteria which will require NDE data processing functionalities developed and demonstrated under this program.

Three case studies were selected and conducted during the development of the package. The first case study applies to the development of a low cost manufacturing process (induction welding of thermoplastic) for an Advanced Structures composite wingbox. Full waveform ultrasonic data, geometry model data, in–process MAUS ultrasonic data and thermocouple data, were fused in a visualization application. The second case study examined the effect on radar

performance of repainting of the E–3 AWACS radome using microwave, ultrasonic data, geometry model data, and radar test range data. In the third case study, AUSS ultrasonic data from B1–B weapons bay doors were fused with HRTRR digital radiographs and a complete set of geometry models.

The package was implemented on a purchased Silicon Graphics workstation which was used for the case studies and was delivered to the Air Force at the conclusion of the contract. The results of the effort were presented to both industry and Air Force personnel in a series of meetings, demonstrations, and software installations.

The program concluded that NDE data fusion is applicable and/or cost beneficial in certain areas. In addition to supporting high value component evaluations, it has a major role in developmental programs, in failure investigations, in NDE science, and in the evaluation of complex monolithic structures. The drive to affordable manufacturing is leading to an increase in the development and production of complex, monolithic structures requiring NDE data fusion functionalities which have been developed and demonstrated under this program. NDE data fusion technology is increasingly becoming a key requirement in these development programs, and is expected to be utilized during production. NDE data fusion will be applied to post–deployment and maintenance scenarios such as the AWACS radome refurbishment where mission criticality is an issue. Low observable signature assurance is a particular area which falls into this category.

Visual programming and object visualization is a widely applicable technology, appropriate for general use by the NDE community. The implementation of INDERS 3 under this program provides a model for expected natural evolution from image processing paradigm to object visualization paradigm.

1.0 INTRODUCTION

Boeing is pleased to submit this final report on work performed under contract F33615–95–C–5234 entitled "Nondestructive Evaluation (NDE) Data Fusion." Our primary goal on this contract has been to upgrade and enhance well established NDE data fusion technology developed on prior government and industry programs, and to transfer data fusion capability to NDE engineers working in government and industry. This technology has been embedded in a software package delivered to the Air Force together with a prototype NDE data fusion computer workstation. The software package is entitled INDERS version 3 and is provided with this report on CDROM. The workstation, a Silicon Graphics Solid Impact running IRIX 6.2, includes all supporting software and licenses used to develop and to operate INDERS Version 3. A photograph of the NDE Data Fusion workstation is shown in Figure 1.0–1.

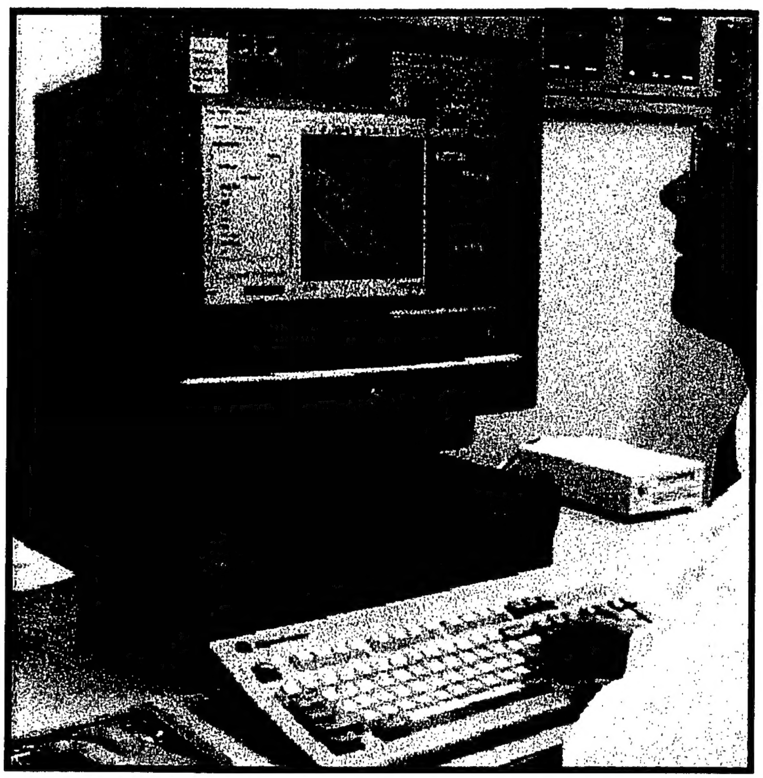


Figure 1.0–1 Photograph of the NDE Data Fusion Workstation

1.1 NDE DATA FUSION ISSUES

Data fusion is defined as "the synergistic use of information from multiple sources in order to assist in the overall understanding of a phenomenon"[1]. Most commonly, the term has been

applied to mixing of image and/or sensor data from scene or target recognition systems on isolated platforms. For instance, airborne or orbital surveillance platforms such as missiles or military satellites often need image data fusion for data compression and/or fast target recognition. The sensor and/or imaging systems are well defined and integrated with the fusion algorithms. In these cases, the object of data fusion algorithms is to reduce the processing time or storage requirement associated with the extraction of information contained in the data.

An example of image mixing is shown in Figure 1.1–1. In this example, wavelet transform methods have been used to combine a forward looking infrared (FLIR) image and a video image of an F–15 taking off fromSt. Louis' Lambert Field. The combined image contains important features of both independent images. By combining images of this type viewed from a common reference frame, the workload on a scene recognition processing system can be reduced, and consequently the time for making critical mission decisions is reduced.

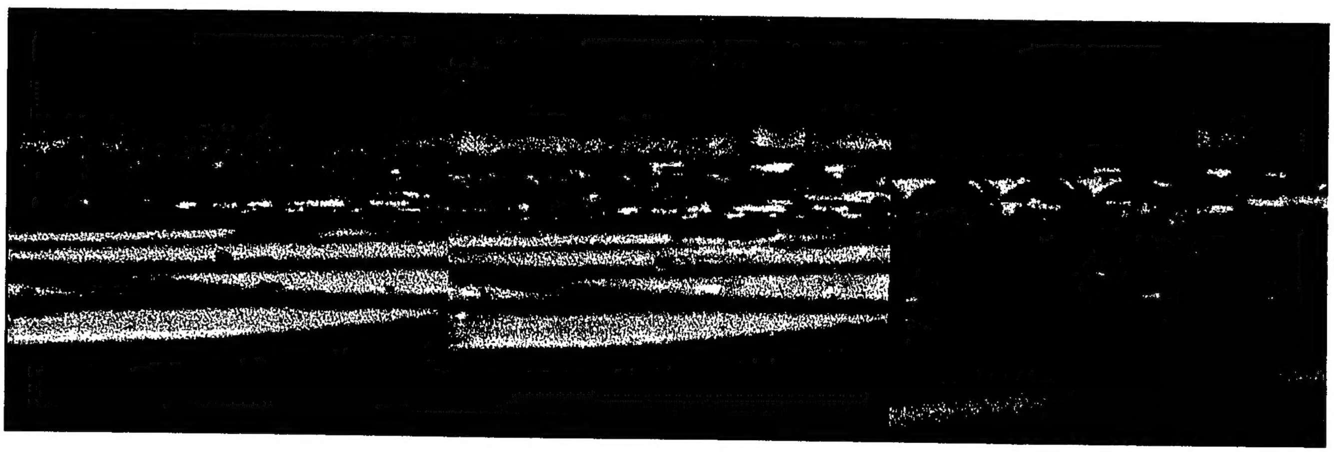


Figure 1.1–1 Fusing of Video and FLIR Images of F–15 Over Lambert Field

Image compression and fast target (i. e. defect) recognition are also desirable features of good NDE data fusion algorithms. However, NDE data fusion algorithms must provide additional functionalities which are unique. Except for the case of colocated camera based inspection systems, NDE data fusion must take into account the geometry of both the inspection systems as well as the part being inspected to achieve correct registration of data. In addition, the time related priorities of NDE data fusion processes are different. Although parts may be available for inspection over short intervals, these are often widely spaced in time and physical location. Consequently, NDE data fusion is often a postprocessing rather than a real–time data processing

step. In a postprocessing scenario, the need to provide unambiguous, "provable" conclusions often is more important than reducing data or image processing workloads. In many cases, data presentations that are acceptable to non–NDE experts such as Manufacturing Review Board personnel or other investigative bodies are required.

1.2 INDERS DATA FUSION METHODOLOGY

To provide data fusion functionality for the unique priorities of NDE data decision processes, not only is image mixing functionality required, but also NDE data processing algorithms which deal with the geometry associated with parts and inspection systems. In mathematical terms, this means being capable of transforming data into Lagrangian (part fixed) systems before combining data. Instead of navigating in an Earth or camera based geometry system, NDE data fusion algorithms must be capable of navigation within a part based geometry system to be effective. Implementation of part fixed coordinate registration is the key functional requirement for NDE data fusion and is the fundamental element of the Integrated NDE Data Evaluation and Reduction System (INDERS) methodology[2].

Part fixed coordinate registration is most easily implemented by utilizing finite element modeling concepts for attaching geometry information to NDE test data. These concepts, described in [2], formed the basis for INDERS 1[6,7,8] and INDERS 2[10], and have been significantly developed and refined in the current software implementation. Figure 1.2–1 illustrates NDE data fusion results from eddy current, ultrasonic, and x–ray computed tomography inspections of an Inertial Upper Stage (IUS) SRM–1 rocket motor nozzle exit cone using INDERS 1 tools.

In the current implementation, the target of NDE data conversion processes is the Unstructured Cell Data (UCD) format[3], an extremely simple ASCII data format standard which supports the finite element model for geometry information . Once NDE data, measurement data, and part geometry data is converted into UCD format, software tools are provided which permit tranformation of data into part fixed coordinates and, consequently, a wide variety of NDE data fusion processing steps. Unlike the prior implementations, the data can also be visualized and transformed dynamically using the computer visualization interface.

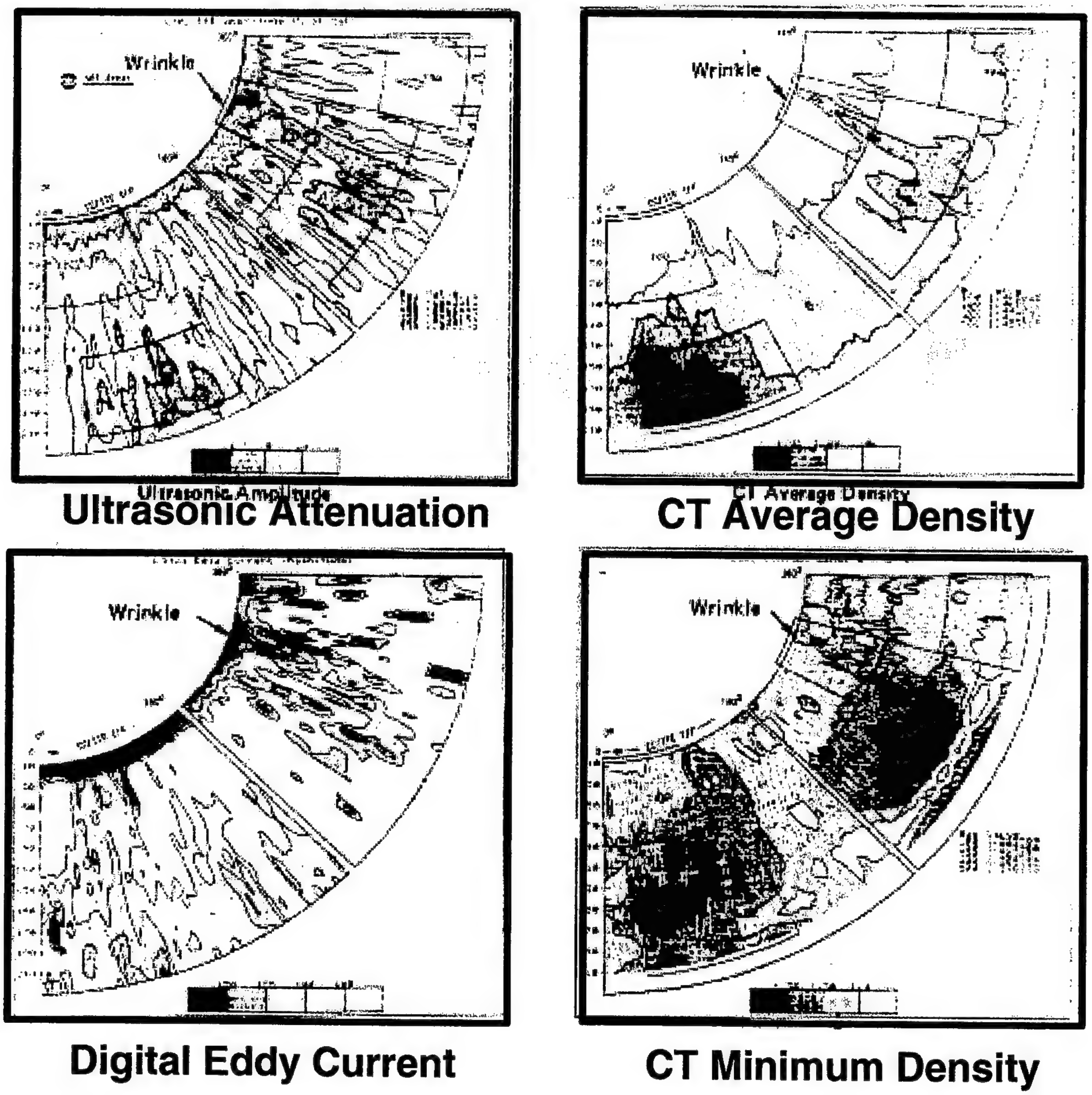


Figure 1.2–1 Example of NDE Data Fusion Results from 1988 IUS Special Study

1.3 VISUALIZATION AND VISUAL PROGRAMMING

Visualization is the art and science of turning complex data into visual insight. Since one third of the human brain is devoted to visual processing, providing data in visual form has the potential to increase comprehension rate significantly. In computer visualization, the power of the human eye and brain are used together with the computer's data processing power to permit rapid comprehension of complex data relationships by presenting data as multi dimensional color images and animations.

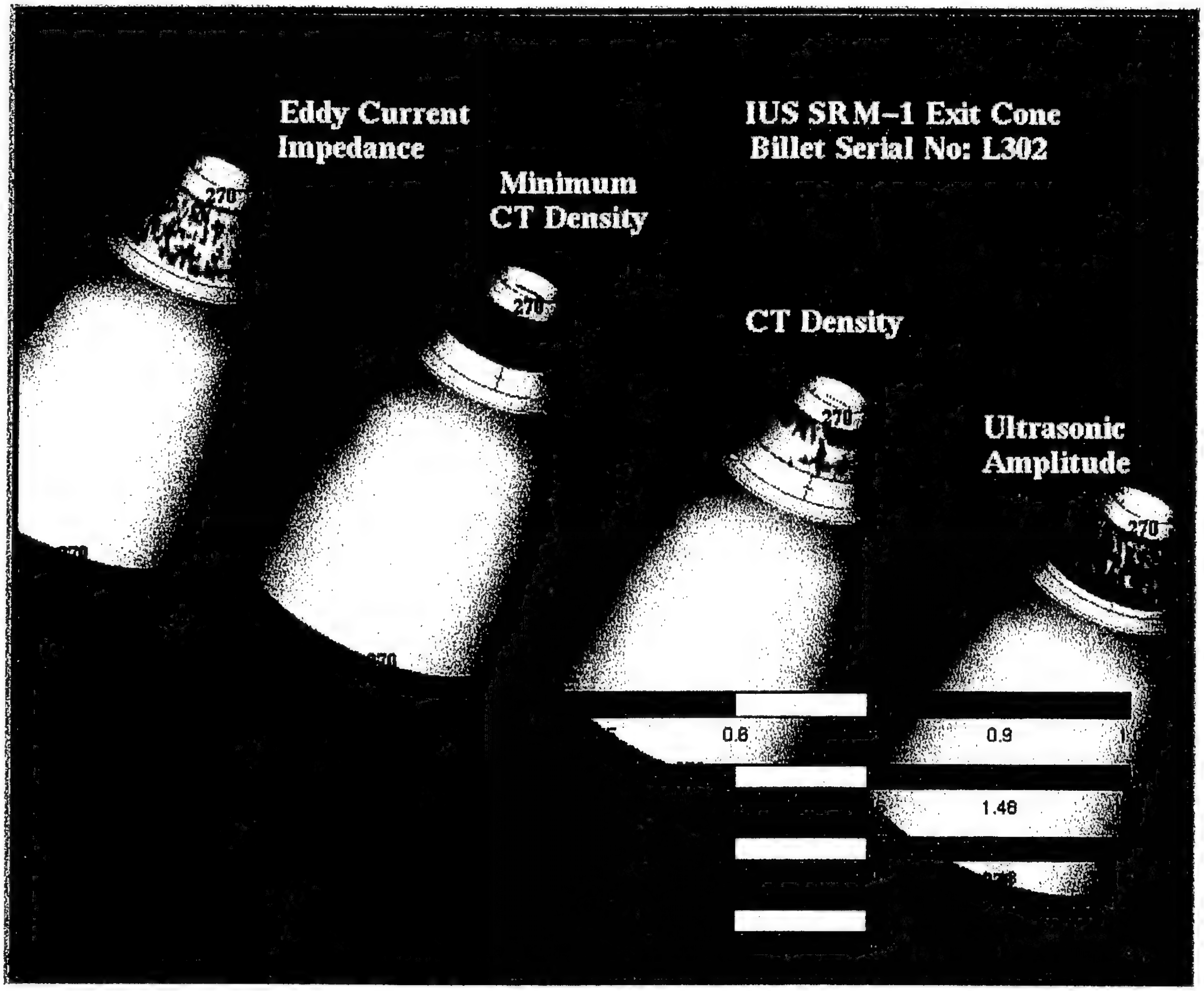


Figure 1.3–1 INDERS 3 Visualization of 1988 IUS NDE Data Fusion Results

Figure 1.3–1 shows the same results for the 1988 IUS example (Figure 1.2–1) as a three dimensional visualization using INDERS 3. Although superficially similar, the latter presentation can be manipulated dynamically using the computer mouse. More importantly, it preserves the actual geometric relationship between the test data and the exit cone shape. A simple question like "how far is that NDE anomaly from the compliance ring?" can be answered as easily as if the actual part were sitting in front of the analyst with the data "chalked on".

The basic implementation strategies behind state–of–the–art computer visualization tools are the finite element modeling concepts discussed in Section 1.2 and used as the basis for INDERS 1[6,7,8] and INDERS 2[10]]. In visualization practice, these concepts have been generalized and extended from the relatively simple methods used in engineering analysis to include such functionalities as texture mapping, light modeling, depth cueing, and high order interpolation

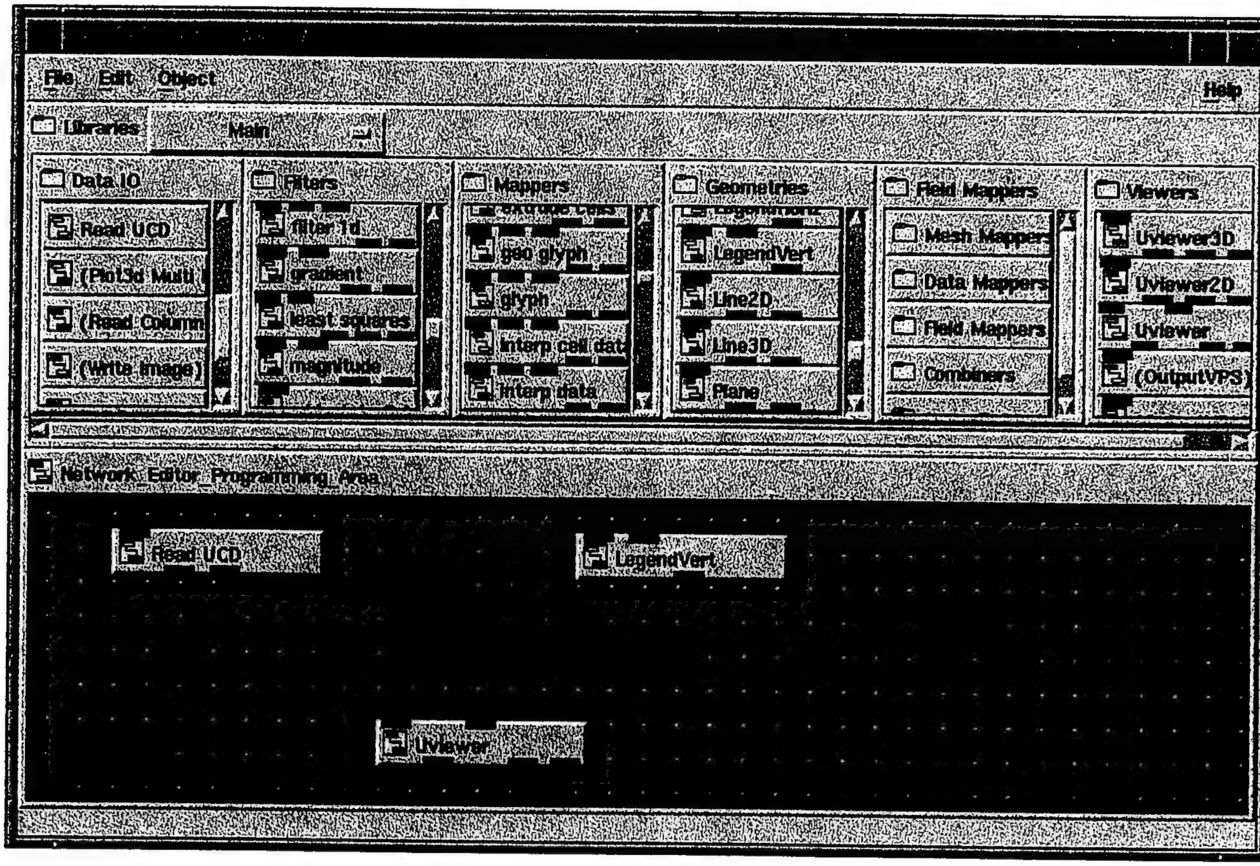


Figure 1.3–2 INDERS 3 Visual Programming Interface aka the Network Editor

elements such as non–uniform rational B–splines (NURBS).

Boeing selected AVS/Express as its visualization platform for INDERS 3. AVS/Express is a multi–platform, component–based software environment for building applications with interactive visualization and graphics features. AVS/Express employs an object–oriented visual programming interface to enable the user to create, modify, and connect application components. In addition, it provides a wealth of fine–grain visual programming objects that provide a complete development environment. High–level objects are available, such as 2D and 3D graphics viewers, data and image processing algorithms, and Graphic User Interface tools.

The visual programming interface, referred to as the Network Editor (NE), is used for programming applications. The Network Editor allows users to construct visualization applications for NDE data fusion as connected, hierarchical networks of objects, with "drag–and–drop" convenience. This approach promotes software reusability and increases programming productivity. With visual programming, a non–expert user can create, modify, and combine

program objects (components) into higher–level application objects. By displaying the hierarchy of objects and their relationships visually, the user is encouraged to use a structured approach to application construction, which dramatically shortens the time it takes to develop, test, and deliver applications.

2.0 PROGRAM DESCRIPTION

The program team consisted of Boeing as the prime contractor, conductor of demonstration case studies, and integrator of the data fusion workstation hardware and software. Perceptics assisted Boeing in performing the requirements analysis and developing the design specification. We used the team's existing Integrated NDE Data Reduction System (INDERS) software package (Versions 1 and 2) as a baseline for this development effort and employed the AVS/Express commercial off–the–shelf application development package as a platform for providing NDE data fusion functionality in a platform independent, visually programmable, 3D visualization environment.

2.1 WORKSTATION DEFINITION

Government and industry members of the JANNAF NDE subcommittee met at the 1992 JANNAF NDE Data Fusion Workshop. The meeting resulted in a draft recommendation for NDE data fusion technology development by the Air Force. This recommendation formed the basis for the workstation definition and subsequent PRDA and Contract SOW. Boeing and Perceptics used a series of usage scenarios to assist in the final definition and prioritization of workstation requirement elements. The workstation requirements are documented in [4]. Key elements are summarized below.

- Develop, integrate, and demonstrate methods for integrating image data from multiple inspection modes
- Deliver a prototype data fusion workstation consisting of image analysis and data fusion software and hardware
- Implement software to translate data to generic format
- Employ COTS image analysis software for workstation
- Reduce barriers to introduction of NDE data fusion technology at AF facilities
- Provide for development of reuseable tools
- Provide a process for developing one–time–use tools quickly

Among the key conclusions of the requirements analysis was the recognition that "NDE Data Fusion is ... an evolutionary process in which new requirements arise in response to new or unusual problems. For this reason, case studies will be performed during the development of the

workstation"[4]. This recognition led to a "toolbox" design concept rather than an "end product" design concept for the software component of the workstation. In this design concept, the workstation is used as a factory for manufacturing NDE applications, suitable for both rapid prototyping of new applications and running of previously developed applications. This approach relies on modern object oriented programming (OOP) concepts which have replaced traditional (i. e. development of FORTRAN subroutine libraries) approach to providing reusable software tools. The associated workstation design is outlined in [5]. Key elements of the design are summarized below.

- Hardware platform: SGI/IRIX
- Software Platform: AVS/Express
- Software Baseline: INDERS 1 and 2 functionality
- Additional New functionality needed in
    - Data representation
    - Data import/export
    - Field math
    - Visual registration
    - Evaluate/solve
    - Dimension and unit handling
    - Batch execution support

The NDE Data Fusion Workstation Design Review was held at Boeing in Seattle in May 1996 to review Boeing's recommended draft system design. The final System Design Document was delivered to the Air Force in June 1998.

2.2 WORKSTATION DEVELOPMENT

The Silicon Graphics workstation was delivered to Perceptics in May 1996. Integration of the workstation hardware and purchased software was completed by Perceptics in May and early June 1996.

An unanticipated delay in scheduled incremental funding by the Air Force pushed the program

into hiatus from June 1996 until November 1996. When funding was received, key personnel at Perceptics could not support the software development work due to other program obligations committed to as a result of the incremental funding delay. Boeing began to develop "pathfinder" software components and assemble prototype case study applications, while awaiting availability of Perceptics key personnel. In early May 1997, Boeing elected to complete the software development work without Perceptics support rather than risk additional program delays. The development workstation was shipped from Perceptics to Boeing on 5/11/97. Additional Boeing personnel were drafted to support the remainder of the workstation development effort.

The first prototype INDERS 3 application was developed for Boeing's Advanced Structures wingbox thermoplastic weld and utilized via internet in the Advanced Structures area. The Advanced Stuctures area is at the Berkeley Site, about 12 miles north of the Boeing Kent Space Center where the developmental workstation resides. This application provided capability to process and review ultrasonic data from the Boeing (full waveform) blade/fillet ultrasonic system, MAUS ultrasonic data, and thermoplastic weld thermocouple data. The data are mapped to three dimensional geometry models of wingbox components. The application allows the operator to examine and modify the ultrasonic signal feature extraction steps. The development workstation hosted application was implemented as a clickable icon in the program area Windows–95 platform using Boeing networks and Exceed 5.0 as the OpenGL emulator.

In the Advanced Structures application, the four types of data were fused into an integrated three dimensional visualization application operating on the AVS/Express visualization platform. Some of the data logistics and conversion for the application was done outside the AVS/Express environment using INDERS I tools, and other work–arounds were utilized due to the nonavailability of planned design components at the time of this application development.

When software component development responsibility was shifted to Boeing, the application data reduction and analysis steps were incorporated into AVS/Express modules. Several pathfinder applications incorporating these components were built, compiled, and tested. The development

of the Boeing case study application for the Advanced Structures wingbox weld was completed in early June 1997.

The Interim Program Review was held at Boeing on 6/17/97. After the review, the Advanced Structures application and Boeing's current INDERS 3 application development environment for NDE data reduction and analysis were demonstrated to Air Force technical personnel (Charles Buynak and Laura Mann). The INDERS 3 application development environment and the Advanced Structures application was provided to the Air Force representatives on CD–ROM.

The funding based delay in initiating development of the software component library and the consequent initial use of INDERS 1 tools and work–arounds to prototype the first case study application provided a fortuitous benefit to the program. In particular, it raised an issue which had not been explicitly considered during system design, which resulted in a significant change to the development approach for selected software components.

It was noted that data format conversions (and certain other functionalities) should be accessible without necessarily requiring a visualization interface or an AVS/Express license. For instance, the Advanced Structures program staff had a high desire to convert the blade/fillet full waveform ultrasonic data from its unique "homebrew" format to industry standard DRUS (Digital Recording for Ultrasonic Test) format for archival purposes. However, it was unreasonable to tie up the limited number of shared AVS/Express licenses for the many hours typically required to perform this conversion for the very large wingbox test data sets. Similarly, ultrasonic signal processing was a time consuming non–visualization activity which unnecessarily tied up limited license resources. Also, there was a desire to perform the in–process data conversions on non–visualization platforms near or at the test facility. Purchase of an additional graphics workstation and/or an AVS/Express license purely for this purpose would be an unnecessary expense.

The solution to this problem was simply to reorganize each of the appropriate software components into two parts, (a) an AVS/Express component containing the GUI interface for

launching the conversion, and (b) a stand–alone program which did not require an AVS/Express license to perform the conversion or signal processing step. The stand–alone program could be invoked from either inside or outside the AVS/Express environment. A consequence of this approach was to increase the importance of having a file based equivalent to the AVS/Express unstructured field visual programming object. The Unstructured Cell Data (UCD) file format [3] met this requirement. Objects for both reading and writing UCD files (ReadUCD and WriteUCD) are provided in AVS/Express. UCD is a human readable ASCII format for NDE data, including the "finite element" geometry definition which provides a convenient baseline for training of NDE engineers in the use of unstructured fields and for debugging of AVS/Express applications.

An additional revision to the design approach was to delete Visual Parse as the software platform for implementing mathematical expression evaluation for "field_math" and "evaluate" functionality [5] in favor of AVS/Express' native language, V. This was a result of gaining familiarity with the V language, and the discovery that it could be augmented readily by adding user defined functionality.

With these revisions to the design approach, the remainder of the software development effort was completed between July 1997 and March 1998. During this period the AWACS radome case study was completed and the B–1B weapons bay door case study was initiated. The specific software components developed and their functionality is described in Section 3.0.
An intermediate release of INDERS 3 on CD–ROM (containing the AWACS application components) was completed on 7/30/97 and provided to the Air Force technical monitor (Laura Mann). A complete executable INDERS 3 CDROM compiled for the NDE Data Fusion workstation was provided to the Air Force prior to the NDE Data Fusion demonstration/workshop at the JANNAF NDE subcommittee meeting on 3/17/98 at the Best Western Olympus Hotel in Salt Lake City. Additional CDROMs were distributed to industry and government personnel at the JANNAF meeting.

In addition, a "beta test" CDROM for a Windows NT version of INDERS 3.1 was completed prior

3.0 NDE DATA FUSION WORKSTATION

3.1 Background and Needs

Practical NDE data fusion requirements encompass great variety. Variations occur in data types, formats, interpretation, dimensionality, operator sophistication, and throughput requirements. Each of these parameters affects the design and/or selection of data management tools and the potential for justifiable return on investment. To be of greatest benefit, a data fusion workstation had to be developed which was suitable for locating in engineering, manufacturing, operations, and maintenance facilities. The workstation hardware needed to be compatible with the facility requirements of the operations environment, including space available, power, and cleanliness.

In designing the user interfaces, the workstation and associated software needed to suit a variety of skill levels corresponding to the personnel who are to use the workstation. In this way, the workstation could be seen as an asset and not a hindrance. For routine NDE personnel use, simplicity and robustness were desired. For certain engineering and/or investigative uses, access to complex tools and a programming environment would be required. The user interfaces had to be intuitive to learn, providing tools that solve the real problems conveniently. Visualization tools were necessary to enhance intuitive comprehension of complex data relationships.

Often–repeated mechanical steps needed to be automated. Finally, the workstation needed to be capable of performing its functions in a timely fashion, consistent with engineering manufacturing, operations and maintenance needs.

There were two major problems identified which are associated with the effective use of NDE data. The first is the lack of a systematic approach to data acquisition, interpretation, and analysis, and the second is the high cost of human interaction and judgment on the very high volume of data that current and emerging NDE techniques generate. To address the lack of systematic approaches to NDE data interpretation and analysis, Boeing had developed a methodology for NDE data interpretation[2] and a large collection of software tools for performing

NDE data reduction (INDERS 1). To address the high cost of human interaction with the large volumes of NDE data produced by advanced techniques, Boeing adopted and extended the visual programming and visualization approaches initiated by Perceptics for the INDERS 2 development program[11]. Previously, the visualization approach had been judged by Boeing to be incompatible with the graphical performance of generally utilized NDE computing systems. However, currently delivered PC based computing systems now can do visualization and this capability is expected to become ubiquitous within 2–3 years.

Most newer NDE hardware is "fully functional" digital–based equipment, which means that ready access to measurement data is available to the data fusion workstation. Those systems often create digital images or data files which require interpretation, normalization, and registration software to be written to convert them to the workstation standard formats. In addition, associated process data may be tabular information, such as spreadsheet files.

The INDERS based data fusion workstation handles the information from these varied digital representations in a manner analogous to a spreadsheet program. A spreadsheet program simultaneously satisfies needs ranging from solving quick arithmetic problems to complex financial computation to database operations to scientific simulation. It does this successfully, because it provides (1) a way of viewing and operating on data that is intuitive and easily understood, (2) a powerful set of simply–accessed tools, and (3) a means for building higher–level solutions whose inner details can be ignored by solution users. Like a spreadsheet, simple tools can be used for ad hoc reading, interactive viewing, and processing and storing data from a large number of modes. These tools can be used interactively for purposes of diagnostic exploration or technique development, or they can be combined in permanently–storable solutions, to be later recalled and reused as though they were part of the original system functionality.

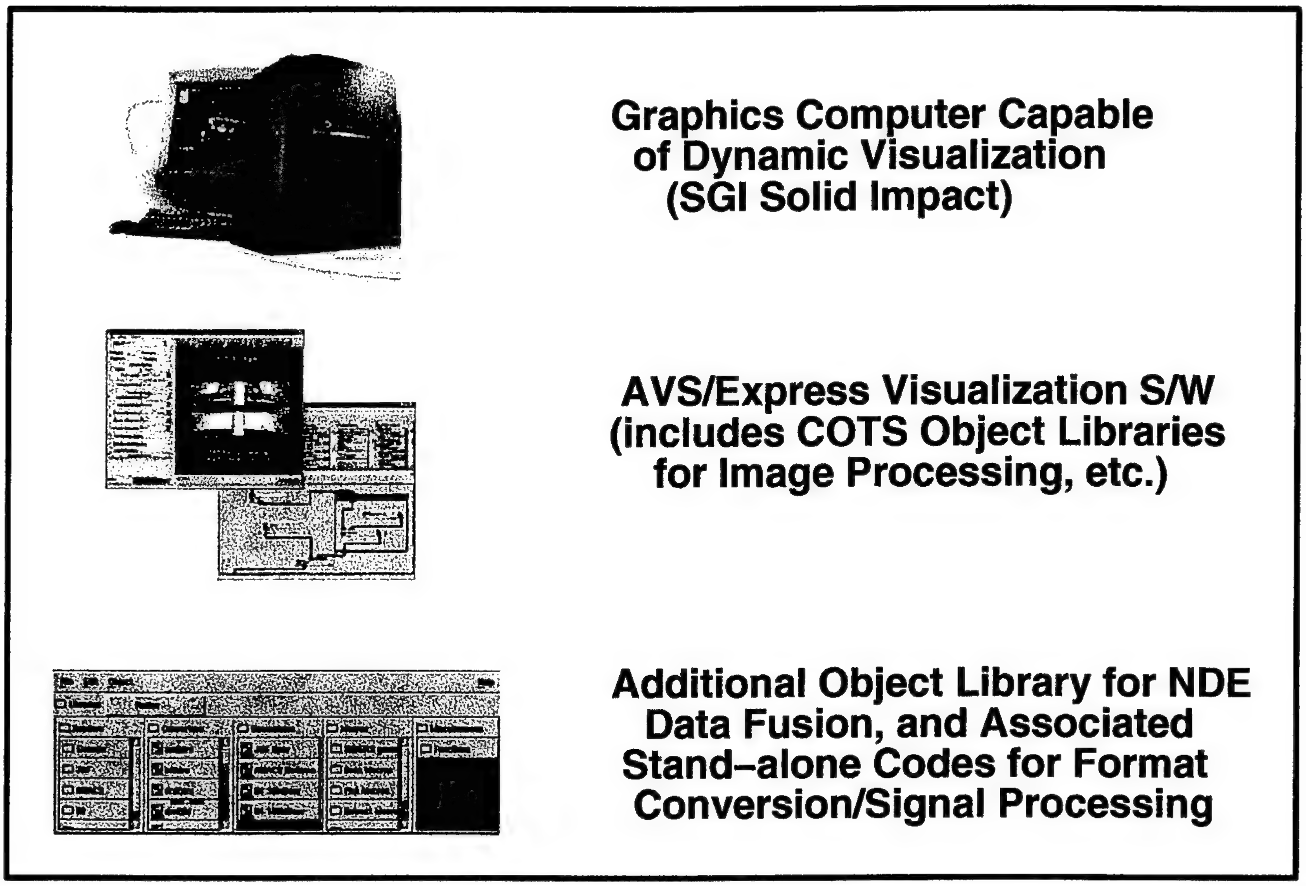


Figure 3.2–1 The NDE Data Fusion Workstation Consists of Three Parts

3.2 Workstation Implementation

The data fusion workstation consistes of three parts as shown in Figure 3.2–1. The workstation hardware platform selected for the program development effort was the Silicon Graphics Solid Impact/R10000 model, as shown in Figure 1.0–1. The features described in Section 3.1 are implemented by layering INDERS on a powerful commercial product: AVS Express™, a product of Advanced Visualization Systems, Inc. AVS Express™ is an object oriented scientific visualization tool that enables data visualization applications to be programmed graphically. [4] Each AVS Express™ module performs some specific task (such as an image processing or visualization operation). Data flows between modules over connecting paths drawn in a click–and–drag style by the programmer/user. The network, its interactive controls, and its displayed images are presented in windows. Once a network solution has been created, it and others can be arranged in menus, and the underlying network can be hidden, making a turnkey application.

Specific NDE data fusion functionality has been added to an existing commercial off the shelf (COTS) product, (AVS/Express™). This added functionality includes NDE data format convertors, highly generalized ultrasonic signal feature extractors, part geometry modelling tools, and data display objects containing part–fixed coordinate transformation functionality. The result is a complete, integrated NDE data processing environment which not only supports NDE data fusion application development but also can be used for general purpose signal and image data analysis as well as visualization.

A functional description of the workstation hardware requirements and software functionality is provided in Appendices A and B. The workstation software (INDERS 3) will operate on a variety of workstations including Silicon Graphics workstations running IRIX and PC workstations running Windows 95 or Windows NT with at least 32Mb of memory.

The Air Force's NDE data fusion workstation developed under this program has been replicated at Boeing to support ongoing NDE data fusion and NDE data analysis tasks. Currently, the NDE group in Boeing's Information, Space, and Defense Systems Research and Engineering (now part of Phantom Works) has seven AVS/Express licenses. Three of our engineers are highly experienced AVS/Express developers. The remainder of the group use the INDERS 3.1 end product in their daily work. Since this has become our primary data analysis platform, we also are providing custom applications to a small number of end users elsewhere within the company.

The workstation was demonstrated to government and industry at the 3/16–20/98 Joint Army Navy NASA Air Force (JANNAF) Technical subcommittee meeting in Salt Lake City recently. Presentations, a hands on tutorial, and the runtime package were distributed on CDROM.

4.0 CASE STUDIES

Three case studies were selected and conducted during the development of the workstation. The first case study applies to the development of a low cost manufacturing process (induction welding of thermoplastic) for the Advanced Structures composite wingbox. Full waveform ultrasonic data, geometry model data, in–process MAUS ultrasonic data and thermocouple data, were fused in a visualization application. The second case study examined the effect on radar performance of maintenance and repair of the E–3 AWACS radome using microwave, ultrasonic data, geometry model data, and radar test range data. In the third case study, AUSS ultrasonic data for the eight B1–B weapons bay doors were fused with HRTRR digital radiographs and a complete set of geometry models. Figure 4.0–1 shows the aircraft for which these studies apply.

These case studies were performed in order to assure that the workstation development would be responsive to the individual needs of the end users. Boeing intentionally required these case studies and workstation development to occur concurrently, with significant feedback between the two efforts. Boeing also presented the results of the effort to both industry and Air Force personnel in a series of meetings, demonstrations, and software installations.

Additional applications of the software were developed during the program period of performance to support ongoing Boeing programs. While these were not funded under this contract, they provided additional insight into user requirements, application requirements, and implementation strategies which are reported in this document under the heading of "Spin–off Applications" (Section 4.4).

Figure 4.0–1 Photographs of Aircraft Involved in NDE Data Fusion Case Studies

4.1 ADVANCED STRUCTURES WINGBOX WELD PROCESS DEVELOPMENT

A pathfinder INDERS 3 application was developed for the Advanced Structures wingbox weld case study and implemented by networking from the development NDE data fusion workstation to end user workstations in the Advanced Structures development area. The Advanced Structures development area is at Boeing's Berkeley Site, about 12 miles north of the Boeing Kent Space Center. This application permits the review and evaluation of ultrasonic data from completed wingbox components examined on the blade/fillet full waveform ultrasonic scanner in conjunction with the weld process thermocouple data, and in process MAUS data from between induction welding steps. Figure 4.1−1 shows a data fusion visualization of a welded test sample. The display includes visual models of process thermocouple data, weld quality factor extracted from the ultrasonic signals, and the weld interface response, all overlayed on geomtery models of the test sample. The user of the application can call up a full waveform ultrasonic viewer and can revise the signal feature extraction parameters for recalculating full waveform ultrasonic signal feature

Figure 4.1−1 INDERS 3 Application for Advanced Structures Wingbox Weld Process Development

maps. In addition, in process MAUS data and other signal features of the blade/fillet data can be called up and displayed on any of the geometry models. The application was implemented as a clickable icon on a Windows-95 platform using Exceed 5.0 as the OpenGL emulator. This demonstrated the practical implementation of a remotely hosted networked INDERS 3 application, validating a key design element. The network server approach was demontrated to be a practical option for convenent delivery of end-use data fusion applications.

The Advanced Structures team has since chosen to implement INDERS 3 locally on their own graphics and structural and thermal analysis server computer (SGI Origin). The NDE data fusion workstation has become an integral part of Boeing's affordable composites process development strategy.

4.2 AWACS RADOME REPAINTING PROCESS IMPROVEMENT

The AWACS radome repainting case study demonstrated the fusion of radome geometry data, full waveform ultrasonic inspection data, microwave (horn stethoscope) reflectance data, and full field radar range test data. The AWACS radar radome is intentionally manufactured with a non-uniform cross-section intended to optimize transmission for the outgoing radar beam and to minimize the internal reflection of ground echoes from beam sidelobes on to the radar antenna array. A thickness error of as little as a single fiberglass ply in the fiberglass substrate or 0.001" in the painted elastomeric rain erosion coating can cause radomes to fail radar range qualification testing. Paint thickness control is critical to the radome's mission, since it is carefully tuned to minimize ground scatter in the antenna receiver. Precise ultrasonic paint thickness gauging requires constructing the analytic envelope signal (using Hilbert Transform methods) from full waveform data and performing autocorrelation of the analytic envelope signal.

Figure 4.2-1 shows a visualization of an AWACS radome together with the raw ultrasonic waveform data, the ultrasonic thickness derived from the raw data as a color mapping, superimposed microwave reflectance measurements as a panelized color mapping (near the equator of the radome) and selected radar response data as a function of azimuthal angle for a selected frequency and reciever orientation.

Eliminating a single radar range test for the radomes saves approximately 600 manhours currently required for radome testing at the Kent Space Center range. The application provides a method for examining the relationship between paint thickness, substrate configuration, and radar performance. Currently the INDERS 3 application is being used for improving the repainting process. This is illustrated in Figure 4.2-2. This figure shows a difference between the ultrasonic thickness after first strip and repaint which resulted in a radar test failure and the second strip and repaint, which resulted in a radar test pass. The E-3 AWACS and 767 AWACS program

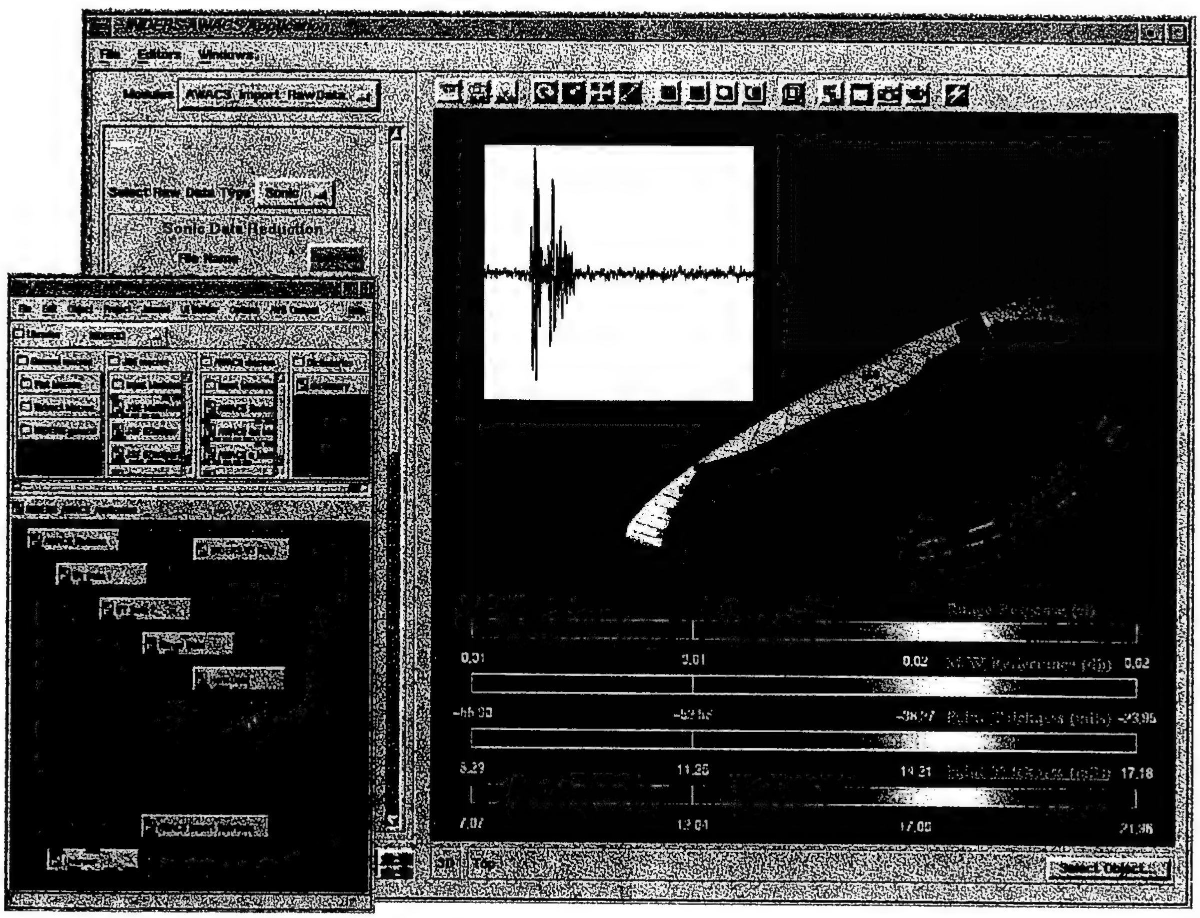


Figure 4.2-1 AWACS Case Study Application Showing Fusion of 4 Modalities

manager (Todd Ray), indicated that these visualizations gave him immediate insight into how to reprogram the painting robot to eliminate the expensive rework associated with stripping, repainting, and retesting on the radar range. This application is being routinely used for both E-3 and 767 AWACS painting and repainting.

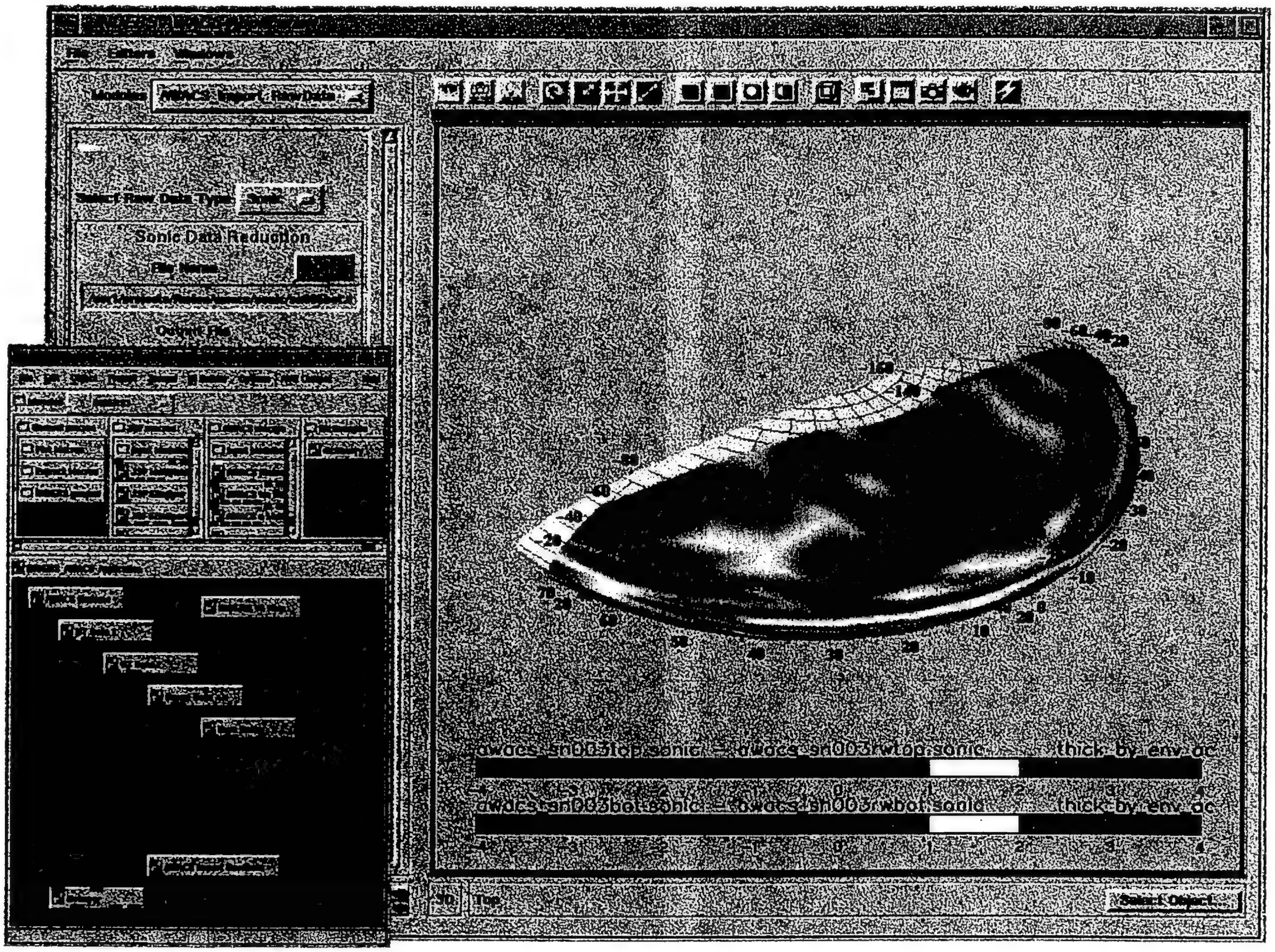


Figure 4.2–2 Visualization of UT Paint Thickness Difference for AWACS Radome Before and After Repainting Corrected Radar Test Performance

4.3 B–1B BOMBER WEAPONS BAY DOOR TEST DATA MANAGEMENT

An application was developed for the B–1B weapons Bay Door NDE data fusion. Each B–1B has eight doors, four 180 inch doors and four 90 inch doors. Figure 4.3–1 is a photograph of the B–1B showing the weapons bay doors. These are ultrasonically inspected by through transmission using the AUSS system at Tinker Air Force Base. Data is also acquired using the HRRTR (High Resolution Radiography) system. However, the HRRTR was not operational at

Tinker, and so data from the HRRTR demonstrations and earlier system implementations was used to develop the data fusion procedures for this application. A series of B-1 Weapons Bay Door digital radiographs from a 1996 HRRTR (High Resolution Radiography) technology demonstrations were acquired from Lockheed Martin for this purpose.

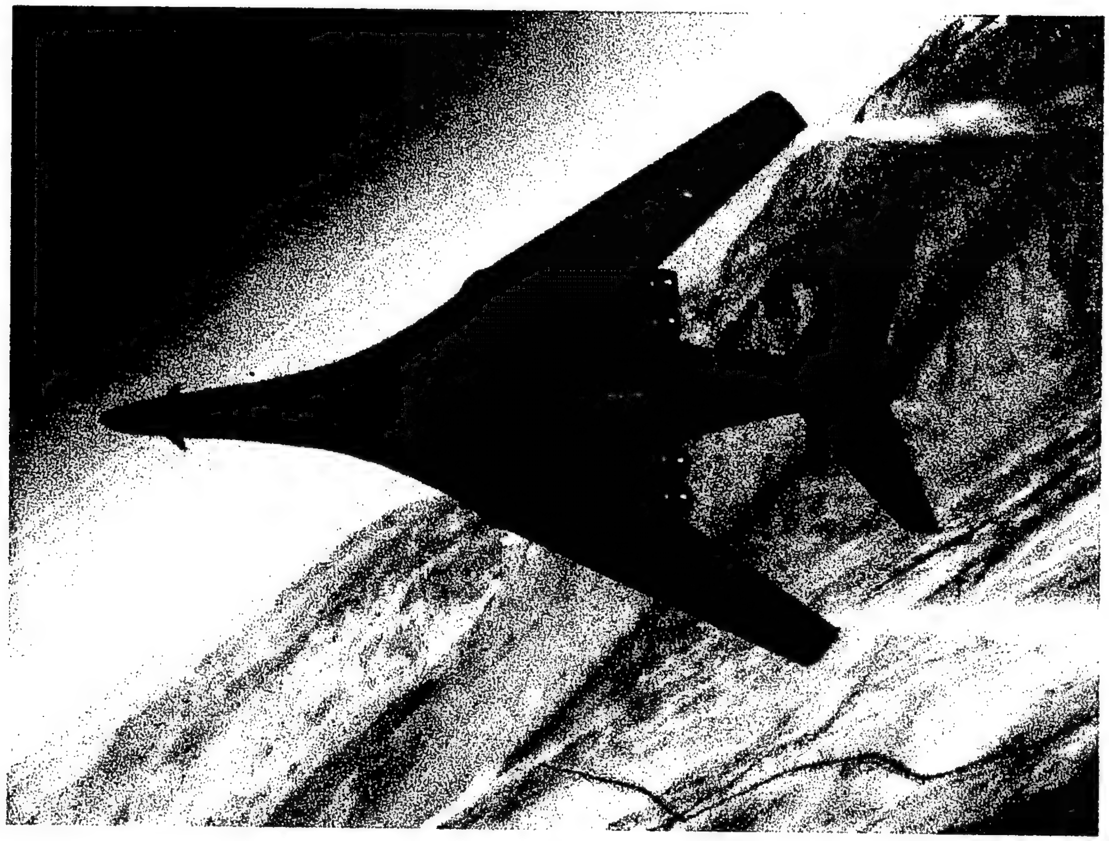


Figure 4.3-1 Photograph of Underside of B-1B Showing Weapons Bay Doors

NASTRAN models for the 90 inch and 180 inch forward and aft B-1 Weapons Bay doors were acquired from Boeing North American, and INDERS 3 geometry models for each of the doors were constructed from them using the methods described in Appendix B Section 5.4. These models contain a complete description of the internal structure of each door as well as the surface skins. The surface skins are visualized as transparent surfaces so the internal details can be visualized. The geometry model for the 180 inch door is shown in Figure 4.3-2.

The ultrasonic test data for the B-1B Weapons Bay Doors was retrieved from the Tinker AFB AUSS system and delivered to Boeing Seattle via Boeing data networks. The data represents 169 AUSS inspections of B-1B doors.

The application permits the rapid review of the ultrasonic data mapped to the surface of the part geometry in a visualization. Flags are attached to the door geometry objects at locations where other NDE data, including HRRTR radiographic images, had been acquired, which can be selected for viewing with a simple mouse click. These are illustrated in Figure 4.3-3. In this visualization, the AUSS through transmssion ultrasonic loss is mapped to the surface of the 180 inch door geometry. Flags on the part (labelled A, B, C, D, and E) indicating the presence of five HRRTR digital radiographs can be selected by the user and the radiographs assembled and viewed in the visualization space. In this case, the lead markers were used to manually register the individual radiographs in the right relative positions by the user, using mouse drag and drop functionality.

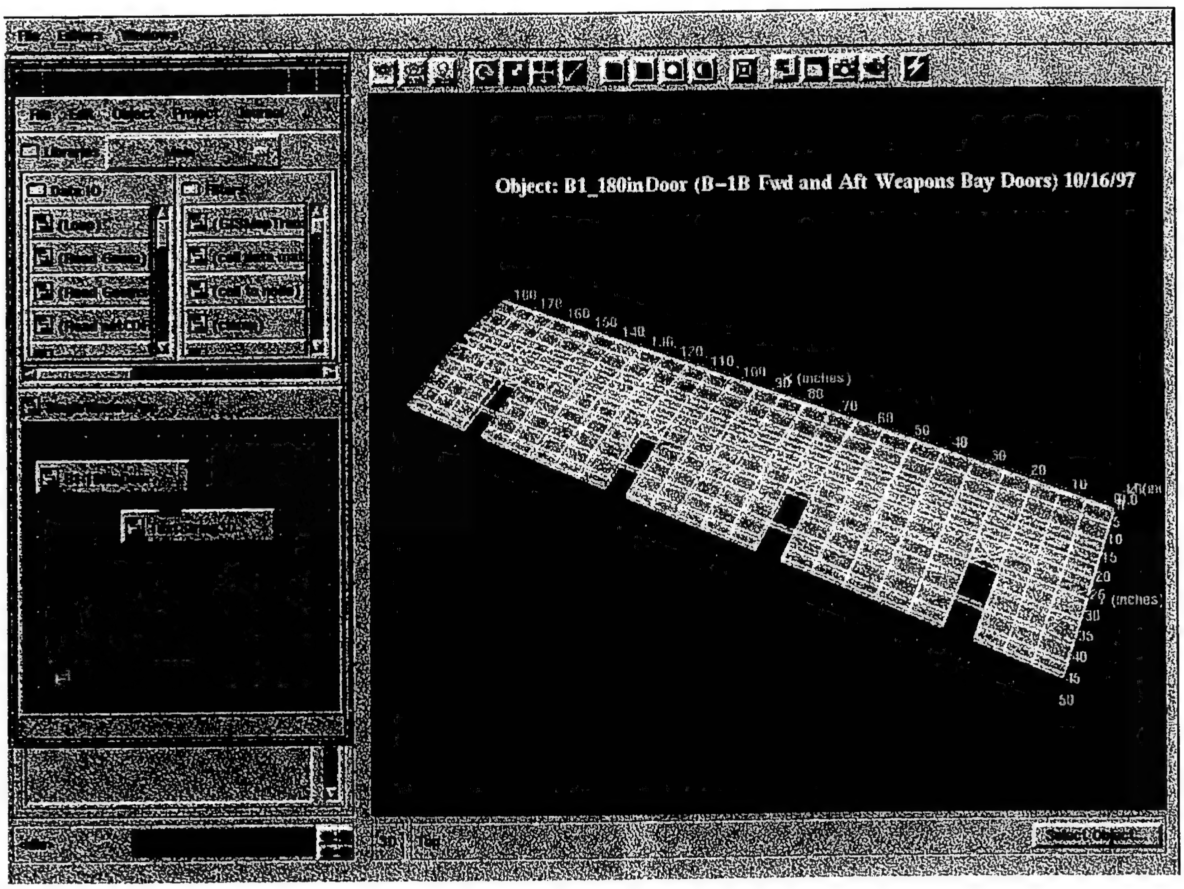


Figure 4.3-2 INDERS 3 Geometry Model of B-1B 180 inch weapons Bay Door

The development of the B-1B case study application was highly beneficial in focussing INDERS 3 development in the area of interactive three dimensional (geometry model constrained) location

selection. However, the application has not been refined or evaluated by the end user (i. e.Tinker AFB personnel) at time of report writing. The acquisition of the AUSS data for the weapons bay doors was delayed until late in the program, by logistical difficulties and, as a consequence, the workstation could not be delivered to Tinker until May 1998.

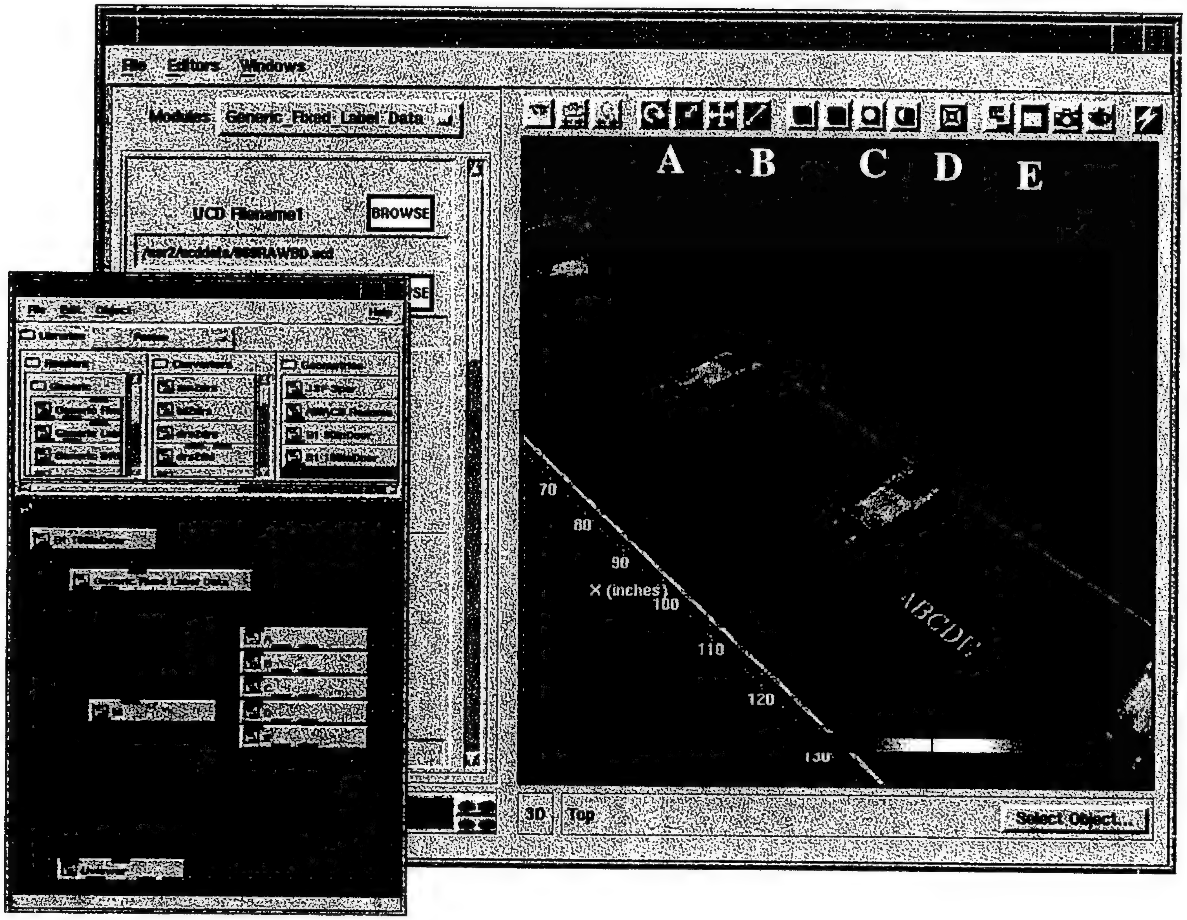


Figure 4.3–3 INDERS 3 Application Fusing HRTRR, AUSS, and NASTRAN Data

4.4 SPIN-OFF APPLICATIONS

During the course of this effort, Boeing has utilized the NDE data fusion workstation on a number of other programs. A large number of applications were developed in support of low cost manufacturing initiatives. In general, these efforts were focussed on fusing of NDE data with geometry models and/or measurement data. In addition, the workstation was used to support radiography simulation, ultrasonic simulation (both conventional and guided wave), NDE data reduction, and a variety of image processing and analysis tasks. Figures 4.4–1 through 4.4–4 illustrate some of these applications.

In Figure 4.4–1, a visualization from a simple application to fuse radiographic images of titanium cast parts with the part geometry derived from the CAD (CATIA) design data. This approach is primarily being used to assist in the prediction of casting radiography capability to designers of aircraft castings. Figure 4.4–2 illustrates the fusing of laser ultrasound data and laser ranging data (both from the LUIS system at Sacramento ALC) for a superplastically formed titanium engine seal for F–22. This data can be used for fabricating chemical milling masks which

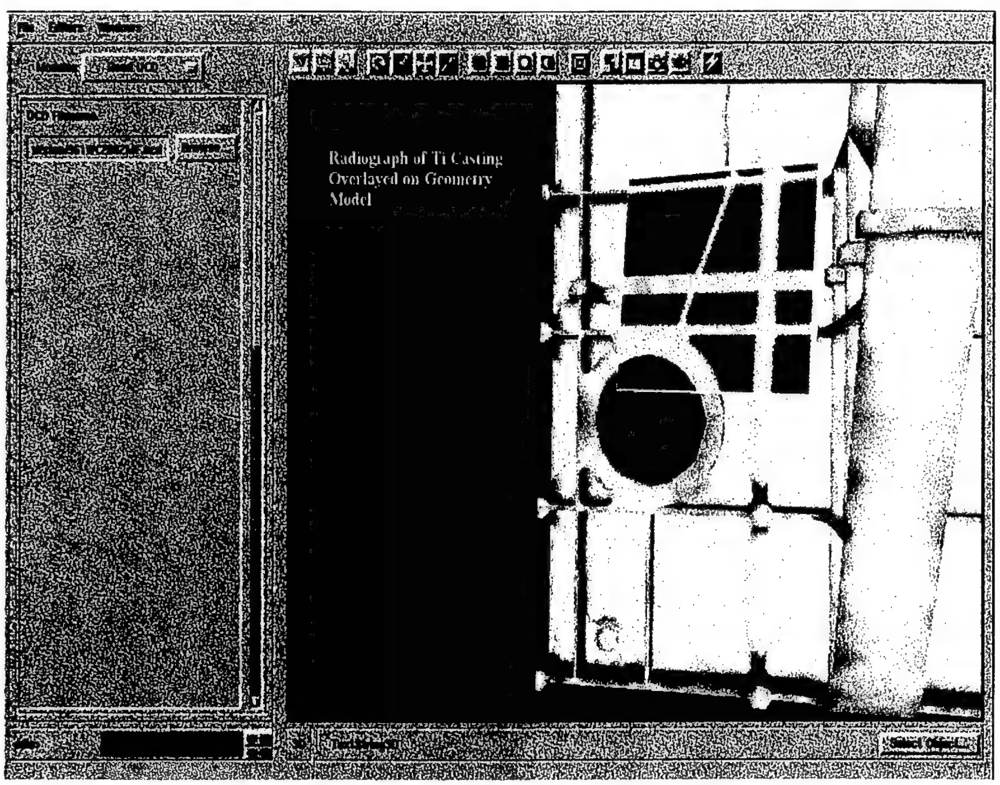


Figure 4.4–1 Fusion of Radiograph and Geometry for Titanium Casting

are subsequently used for precision machining of superplastically formed high performance parts. Figure 4.4–3 shows the fusion of an INDERS 3 CT reverse engineered geometry model with CT derived thickness for a chip from a high speed drilling operation. The chip is a witness item which encodes in its thickness the critical drilling parameters being studied. These are used to calibrate high temperature, high strain rate three dimensional drilling analyses which are used to develop low cost, high speed manufacturing processes.

Figure 4.4–4 illustrates the combination of CT images of the gauge section of a loaded adhesively bonded shear coupon. In this application, images are first co–registered in a Lagrangian sense (i. e. common center of mass and center of rotation), and the coordinates are subtracted to yield a differentiable displacement field, from which strains are calculated. This replaced a stand–alone application which had been very costly to develop without the INDERS 3 tools. INDERS 3 geometry modeling approach and built–in unstructured field operations (i. e. field gradient) allow an analyst to assemble the application in minutes.

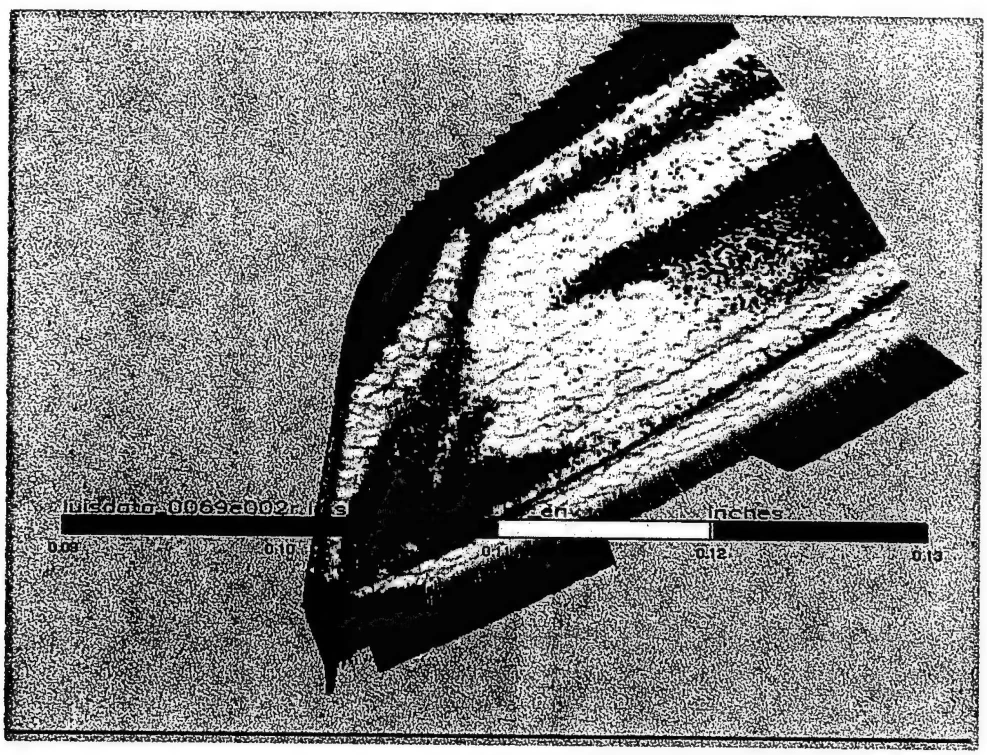


Figure 4.4–2 Fusion of LUIS Ultrasonic Thickness Data with Laser Range Data for Part of F–22 Superplastic Formed Titanium Engine Seal

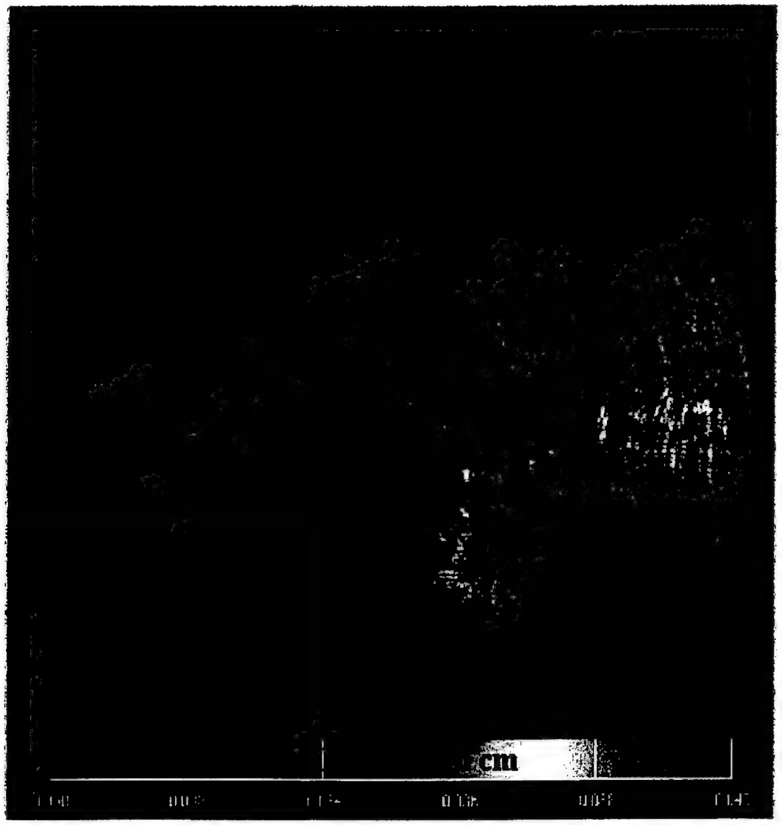


Figure 4.4–3 Fusion of CT Geometry Model with CT Derived Thickness for Manufacturing Development Witness Item (High Speed Drilling Chip)

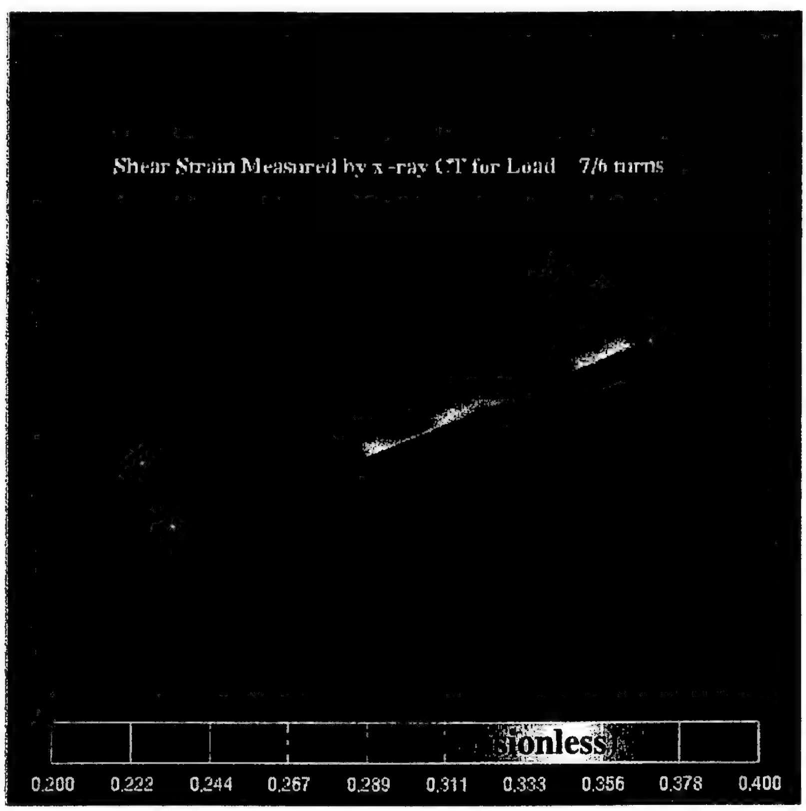


Figure 4.4–4 Fusion of CT Scans from Two Load States to Calculate Shear Strain In Adhesive Bond (High Speed Civil Transport Development)

5.0 CONCLUSIONS AND RECOMMENDATIONS

The program concluded that NDE data fusion is applicable and/or cost beneficial in certain areas. In addition to supporting high value component evaluations, it has a major role in developmental programs, in failure investigations, in NDE science, and in the evaluation of complex monolithic structures. The drive to affordable manufacturing is leading to an increase in the development and production of complex, monolithic structures requiring NDE data fusion functionalities which have been developed and demonstrated under this program. NDE data fusion technology is increasingly becoming a key requirement in these development programs, and is expected to be utilized during production. NDE data fusion will be applied to post–deployment and maintenance scenarios such as the AWACS radome refurbishment where mission criticality is an issue. Low observable signature assurance is a particular area which falls into this category. Because of advances made under this contract, the cost of introducing this technology to program practice has been substantially reduced.

Visual programming and object visualization is a widely applicable technology, appropriate for general use by the NDE community. The implementation of INDERS 3 under this program provides a model for expected natural evolution from image processing paradigm to object visualization paradigm.

6.0 REFERENCES

1. X. E. Gros, NDT Data Fusion, John Wiley & Sons,1997

2. J. M. Nelson, D. W. Cruikshank, and S. Galt, "A Flexible Methodology for Analysis of Multiple–Modality NDE Data", Review of Progress in Quantitative Nondestructive Evaluation, 1989, pp. 819–826

3. Unstructured Cell Data (UCD) File Format, NCSA Tutorial, University of Illinois at Urbana–Champlain,http://www.ncsa.uiuc.edu/Apps/SE/PET/courses/csm–02/lecture01/sld010.hm

4. James M. Nelson and James E. Youngberg, System Requirements Specification for a Nondestructive Evaluation Data Fusion Workstation, 2/10/96

5. David B. Argyle and James E. Youngberg, Software Design Document for a Nondestructive Evaluation Data Fusion Workstation,Boeing Document No. D658–10665–1, 5/2/96

6. Integrated Nondestructive Evaluation Data Reduction System(INDERS) for Nozzle Components Software, Boeing Document No. D180–31159–1.

7. Integrated Nondestructive Evaluation Data Reduction System (INDERS) Software Documentation Package, Submitted To NASA Marshall Space Flight Center under Contract No. NAS9–16960 Schedule 2, 12/22/89, Boeing Document No. D180–31159–2

8. G. E. Georgeson, B. M. Lempriere and J. E. Shrader, "Integrated Nondestructive Evaluation Data Reduction System (INDERS)," JANNAF, Silver Springs, MD, October 1989.

9. Minutes of the JANNAF NDE Subcommittee working group on Multimodal NDE Data Fusion, July 1992

10. INDERS/2 User Manual and Design Reference Version 1, Perceptics Corp.,11/24/93

APPENDIX A

Acronyms and Abbreviations

AWACS	Airborne Warning and Control System
CAD	Computer aided design
CASE	Computer aided software engineering
CDROM	Compact Disk Read Only Memory
COTS	Commercial off–the–shelf software
CT	Computed tomography
CTAD	Wright Laboratories sponsored CT Applications Development program
DF	Data fusion
DR	Digital radiography
EC	Eddy current
EV	Enhanced visual
FE	Finite element
GUI	Graphical user interface
HRRTR	High Resolution Real Time Radiography System developed by Lockheed–Martin
INDERS	Integrated NDE Data Evaluation and Reduction System
IRIX	Silicon Graphics' version of the Unix operating system
JANNAF	Joint Army, Navy, NASA, Air Force
LAN	Local area network
MAUS	Mobile AUtomated Scanner
MM/DF	Multimode data fusion
NDE	Nondestructive Evaluation
STEP	ISO 10303 standard for Computer Aided Design part geometry data
SOTA	State of the art
SRS	System Requirements Specification
UI	User interface
UT	Ultrasonic testing
WAN	Wide area network

APPENDIX B

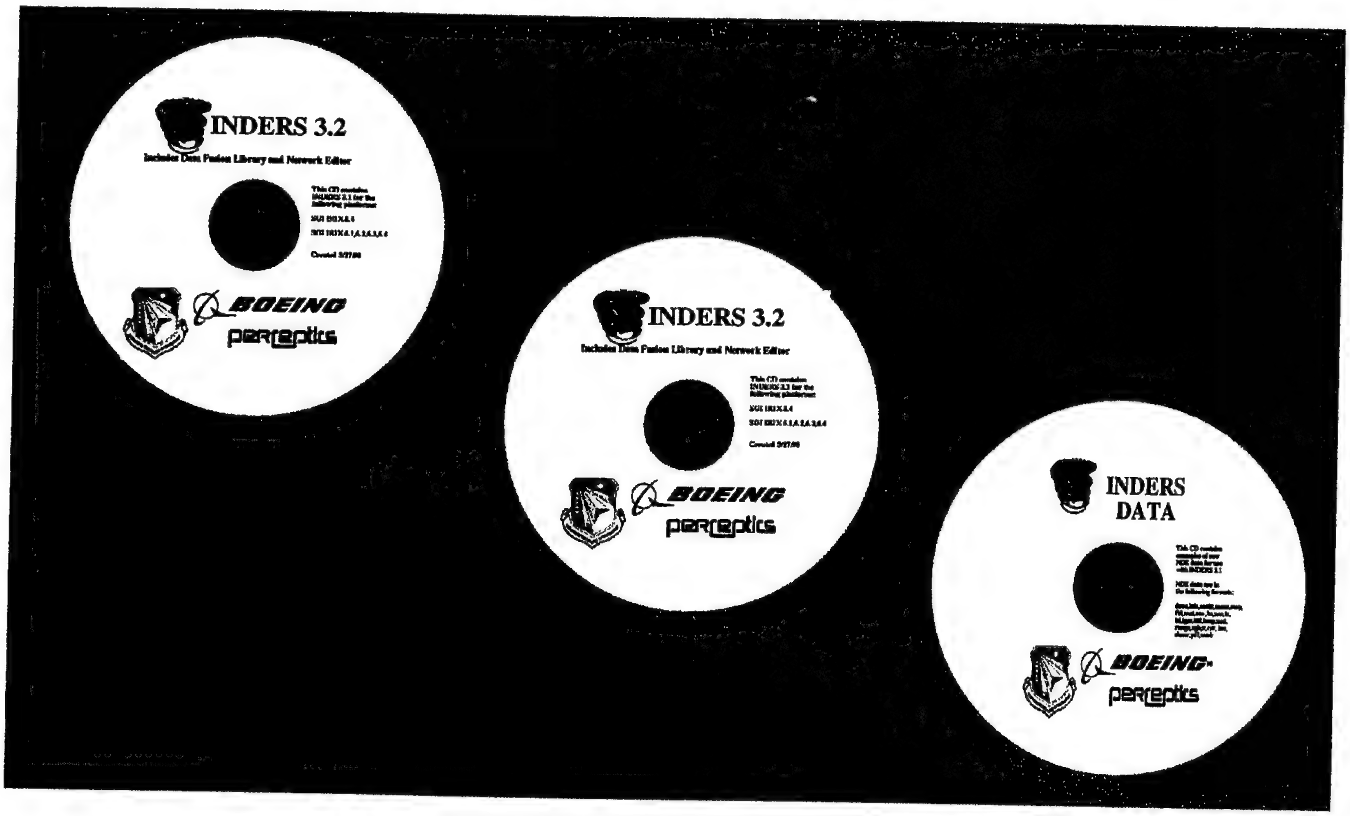
INDERS 3 User's Manual and Software Documentation

INDERS 3

Users Manual

and

Software Documentation



Last Revision 5/26/98

Associated Manuals:	AVS Inc. Part Number
Windows System Prerequisites, Installation, & Licensing	330–0420–04
Installation and Licensing	330–0401–04
Getting Started	320–0313–04
Developer's Reference	320–0311–04
Visualizing Your Data with AVS/Express	320–0320–04
User's Guide	320–0321–04
Data Visualization Kit	320–0314–04
User Interface Kit	320–0315–04
Graphics Display Kit	320–0312–04
Annotation and Graphing Kit	320–0317–04

CONTENTS

1.0 INTRODUCTION

This user's manual is intended to assist new users of INDERS 3 to install the software and to operate it. INDERS 3 is implemented as an extension of AVS/Express[1], a proprietary visually programmable application designed for assembling visualization applications from a variety of powerful software components (referred to as programming objects). These software components are divided into a series of collections (referred to as programming object libraries), which group components with similar or related functionality.

INDERS 3 includes nine of the provided AVS/Express component libraries, and also includes an additional component library with tools containing specific NDE data fusion functionality, named the "Fusion" library. The complete set of components contained in the nine AVS/Express libraries are fully documented in associated online AVS/Express reference manuals which are provided on the INDERS 3 CDROM.

In this manual, only components which are commonly required for NDE data fusion applications will be described in detail. In some cases, this document will reference the relevant AVS/Express online documents[2] by section and paragraph number using the following codes.

AVS/Express Online Document	Reference Code
Annotation and Graphing Kit	AG
Data Visualization Kit	DV
Graphics Display Kit	GD
User Interface Kit	UI

For instance, the component "cell_data_math" [DV5.6] in the "Main" library is used for ...

Section 2 discusses the workstation system requirements for running the INDERS 3 software. Installing an AVS/Express license, which is required for running INDERS 3 is discussed in Section 3. Free 30 day demonstration licenses are available from AVS Inc.. Section 4 provides a series of simple "follow the instructions" tutorials designed to familiarize the new user with the terminology used in both this manual and the documentation, and to provide an intuitive understanding of the visual programming interface, called the "Network Editor". It is strongly recommended that each user work through these tutorials before attempting to develop their own NDE data fusion applications. Section 5 assists the user in selecting the right components for assembling an NDE data fusion application. Section 6 provides instructions for printing AVS/Express online documentation pages conveniently.

1 AVS/Express is a trademark of Advanced Visual Systems.

2 The specific AVS/Express documentation referred to in this document is the version 3.2 documentation set.

2.0 SOFTWARE REQUIREMENTS

To install and run INDERS 3 software you must have either a UNIX workstation running IRIX 5.4, IRIX 6.1, IRIX 6.2, IRIX 6.3, or IRIX 6.4 or a PC based workstation with 64Mb of memory running Windows/NT 4.0. INDERS executes directly from CDROM, so it does not require additional disk space on your system. However, a CDROM drive is essential.

INDERS 3 can be recompiled and configured for a variety of other workstation combinations by an organization such as Boeing who has a special AVS/Express *developer* license.

3.0 LICENSE INSTALLATION

Since INDERS 3 was developed on government contract, no additional licenses are required. Only an AVS/Express license is required to run it. To install the AVS/Express license, you must have system administrator privileges.

There are two types of AVS/Express licenses, a *demonstration* license (free, but only good for 30 days) and a *permanent* license (approximately $2500 for PC and $3500 for Unix system). Installing the demonstration license is as follows:

<u>Installing a demonstration license on Unix systems</u>

(1) Obtain license password from Advanced Visual Systems
call John Hopkins (408) 501-0200 west of the Mississippi
or call Mary Hallice (781) 890-4300 east of the Mississippi

Your license password will be a sequence of characters like A6bf#ft01

(2) The AVS/Express license information needs to be placed in the file:

/usr/local/FLEXlm/licenses/runtimelicense.dat (UNIX systems)

If any directory in this path does not exist, it must be created.

(3) Execute the program "demo_license" from that directory by typing:

```
cd /usr/local/FLEXlm/licenses
/CDROM/express/bin/sgi/demo_license
```

The program will prompt you with

```
license type? [xp|vxp, default xp]
```

Hit return, then the program will prompt you with

```
Generating an Express Developer's Edition license

Password:
```

Type in the password given by Advanced Visual Systems, then hit return twice.

Finally, type:

```
cp license.dat runtimelicense.dat
```

The license is now installed.

<u>Installing a demonstration license on PC systems</u>

(1) Obtain license password from Advanced Visual Systems
call John Hopkins (408) 501-0200 west of the Mississippi
or call Mary Hallice 781-890-4300 east of the Mississippi

Your license password will be a sequence of characters like A6bf#ft01

(2) The AVS/Express license information needs to be placed in the file:

C:\flexlm\license.dat (PC systems)

If any directory in this path does not exist, it must be created.

(3) Execute the program "demo_license" from that directory by double clicking on the icon.

The program will prompt you with

```
license type? [xp|vxp, default xp]
```

Hit return, then the program will prompt you with

```
Generating an Express Developer's Edition license

Password:
```

Type in the password given by Advanced Visual Systems then hit return twice.

The license is now installed.

<u>Installing a permanent license on Unix systems</u>

(1) Obtain the license information (not a password) from Advanced Visual Systems

Your license information will be three lines of information, like:

```
SERVER thrasher 690a0e25 1700
DAEMON avs_lmd /usr/avs/license/avs_lmd
FEATURE XP_RUNTIME avs_lmd 3.000 1-jan-00 1 3CD248592JJ734AAD09E "gvdc" 690a0e25
```

(2) The AVS/Express license information needs to be typed into the file:

/usr/local/FLEXlm/licenses/runtimelicense.dat (for UNIX systems)

If any directory in this path does not exist, it must be created.

(3) On Unix systems the FLEXLM license manager daemons must be initiated and the system rebooted, by typing the following:

```
cd /etc/init.d
cat << EOF > lmgrd
#!/bin/sh
XP_PATH=/CDROM/express
LM_LICENSE_FILE=/usr/local/FLEXlm/licenses/runtimelicense.dat
export XP_PATH LM_LICENSE_FILE
if [ -x $XP_PATH/bin/sgi/lmgrd ]; then
  echo Starting lmgrd
  $XP_PATH/bin/sgi/lmgrd \
  -c ${LM_LICENSE_FILE} \
  -l /tmp/lmgrd.log -2 -p &
  sleep 5
fi
EOF
chmod 755 lmgrd
cd /etd/rc2.d
ln -s /etc/init.d/lmgrd S99lmgrd
/etc/reboot
```

The license is now installed.

To run INDERS 3 for the first time, insert the INDERS 3 CDROM into the CDROM drive. Click on the CDROM icon on your desktop. When the file manager window comes up, click on the file README and follow the instructions. These instructions will set up your home directory and environment for running INDERS 3.

After initial setup, INDERS 3 can be run simply by clicking on the xpr icon in the file manager window.

Installing a permanent license on PC systems

(1) Obtain a license.dat file from Advanced Visual Systems

Your license information will be three lines of information, like:

```
SERVER chinook DISK_SERIAL_NUM=690a0e25 1700
DAEMON avs_lmd C:\EXPRESS3\bin\PC\avs_lmd.exe
FEATURE EXPRESS/PC avs_lmd 2.000 1-jan-00 4 J734AAD09E "gvdc" 690a0e25
```

The server name and directory path names will need to be changed to correspond to your PC system; complete instructions are provided by AVS.

(2) Copy the license.dat file into the directory C:\FLEXlm on your computer. If the directory does not exist, it must be created. If the FLEXlm lmgrd license manager daemon (version 4.1 or higher) has not been installed, it must be copied from the INDERS 3 CDROM and installed on your system. The daemon is located on the INDERS 3 CDROM at the following location:

<CDROM>\express\runtime\bin\pc\lmgrd.exe

(3) Install and start the FLEXlm license manager daemons by opening a (DOS) command shell and typing the following in this window:

Windows NT:

```
INSTALL C:\EXPRESS3\bin\PC\lmgrd.exe
Reboot the computer
```

Windows 95:

```
CD   C:\EXPRESS3\runtime\dlls
COPY *.* C:\Windows\System

CD   C:\EXPRESS3\bin\PC
lmgrd -app
```

The license is now installed.

To run INDERS 3 for the first time, insert the INDERS 3 CDROM into the CDROM drive. Open a Windows Explorer; click on the CDROM icon in the Explorer. When the file contents appear, click on the file READMEPC and follow the instructions. These instructions will direct you to set up your home directory and environment for running INDERS 3.

After initial setup, INDERS 3 can be run simply by navigating to your home directory (with the Explorer) and clicking on the xpr icon in the Contents window.

Other Licensing Approaches

Several other optional licensing approaches can be utilized, including using networked license servers and shared license resources. These are outside the scope of this document. Refer to AVS Inc. licensing documentation part numbers 330-401-04 and 330-420-04 for additional information. These manuals are identified on the cover of this manual.

4.0 USING THE INDERS 3 NETWORK EDITOR

Assembling NDE data analysis applications is done using a visual programming interface referred to as the Network Editor. This interface is described in detail in chapters 2 and 3 of the AVS/Express User's Guide entitled "Working with the Network Editor" which is included in your online documentation.

Before reading the detailed documentation, each new user should work through the following two exercises. These are presented in a "cookbook" approach, where the user simply follows the instructions and observes what occurs on the computer screen. The Network Editor is a very powerful tool, but , as a consequence, it involves a number of concepts which may be unfamiliar to users without significant object oriented programming or visualization modelling experience.

The objective of these exercises is to help the user get an intuitive "feel" for what INDERS 3 programming objects look like, how they are assembled, and where all of the controls for both programming and application operation are located. An additional important element is to teach the user about AVS/Express terminology, which is of great assistance in understanding the online manuals and the remainder of this document.

4.1 First Exercise – Introduction to Unstructured Cell Data (UCD) Readers

In this exercise, the user is introduced to the Network Editor Window, the Application Window, the Visualization Object (*Uviewer*), and three successively more powerful UCD reader objects (*Read_UCD, Generic_Fixed_Label_Data*, and *Radome_Top*). *Read_UCD* is a programming object which provides a graphical user interface (GUI) for the user to select and visualize an unstructured field. The user can think of an unstructured field as a very generalized kind of image, that is, one with three dimensional shape and a named list of components only one of which is displayed at a time.

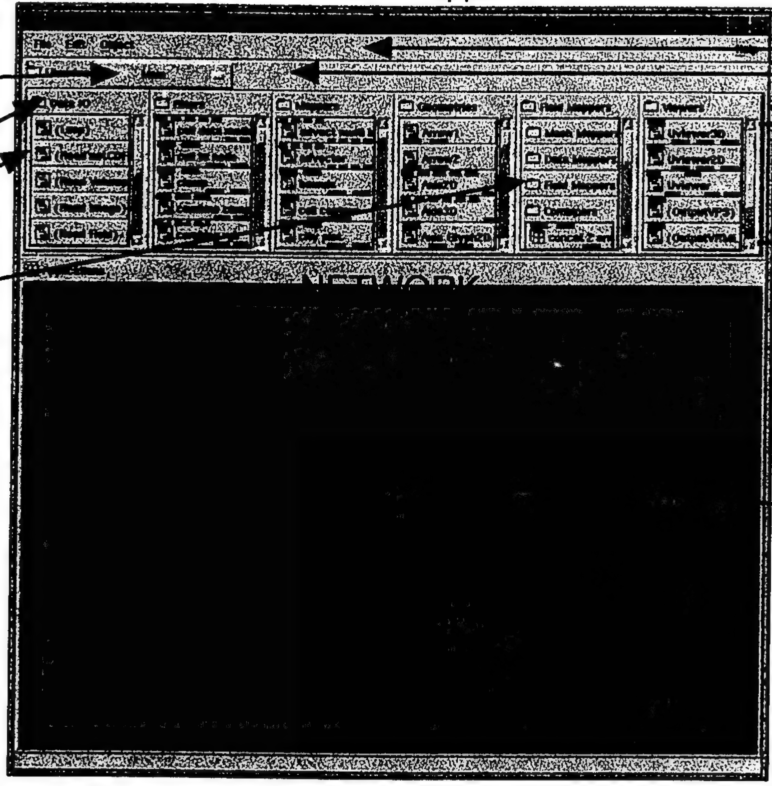
The *Generic_Fixed_Label_Data* is an extension to *Read_UCD* which allows the user to select up to two UCD files for comparison, to select the image component to be displayed, to deform the image (by entering coordinate transform equations), to select and modify the displayed color scale, and to compare the two unstructured fields by means of difference or quotient maps.

Generic_Fixed_Label_Data is a generic tool, designed as the <u>base class</u> for a part specific unstructured field object such as *Radome_Top* . *Radome_Top* is an object of <u>type</u> *Generic_Fixed_Label_Data* which has specific transform equations set to map ultrasonic test data for the top of the AWACS radome to the AWACS radome geometry object (*AWACS_Radome*). An NDE engineer working on the AWACS problem simply tailored a *Generic_Fixed_Label_Data* and then saved it as a *Radome_Top.*

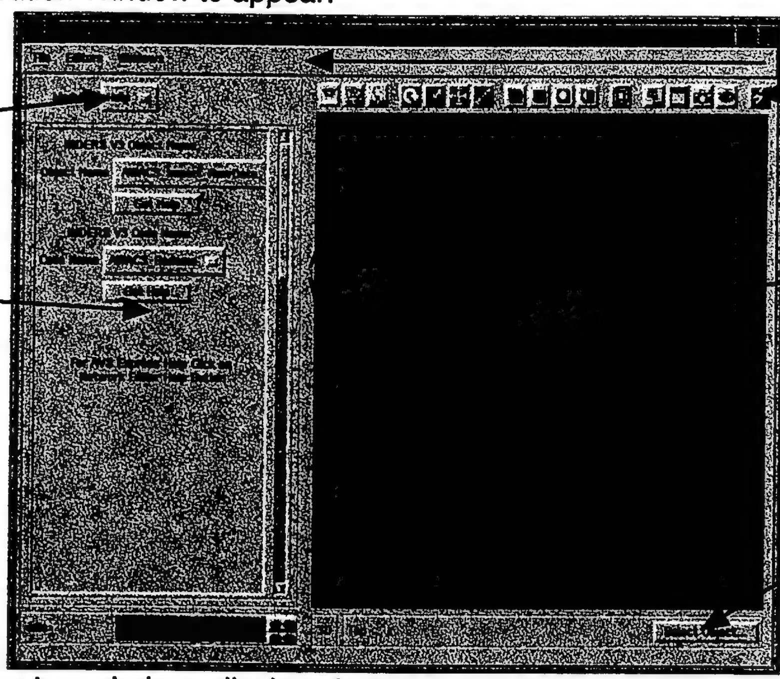
If you are totally confused by the terminology of the last two paragraphs, you are in good company. Object oriented programming (OOP) requires a unique perspective and different terminology from conventional programming. The exercises are designed to make the use of OOP tools easy and intuitive to a nonprogrammer.

INTRO TO UNSTRUCTURED CELL (UCD) DATA READERS

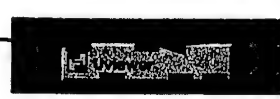
1. ♦ Place the CD entitled "INDERS 3.1" into the CDROM holder, then into the CDROM Drive.
   ♦ Double–click (using the left mouse button) on the CDROM icon on your desktop.
   ♦ Double–click on the "xpr" icon in the CDROM window.
   ♦ Wait 2 minutes for the Network Editor Window to appear:



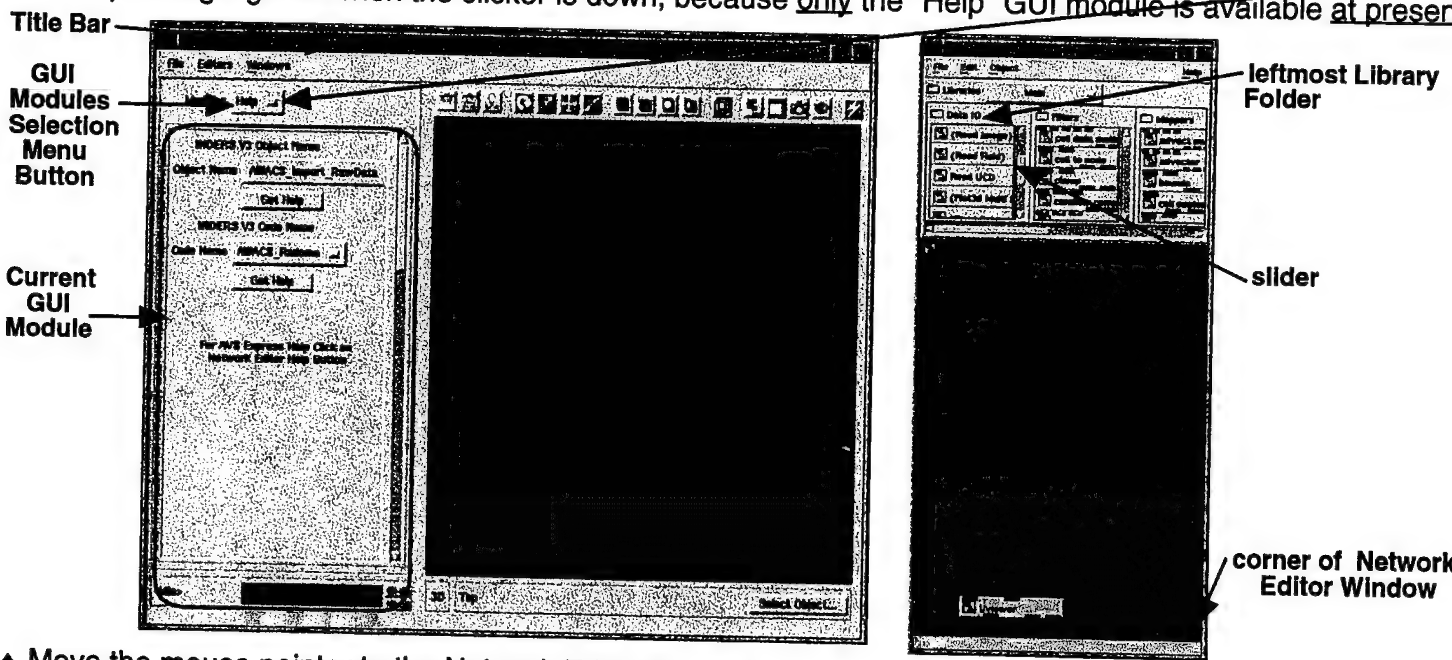
2. ♦ From the Main Network Editor Menu (top lefthand corner), select "File", then "New Application".
   ♦ Wait for the Application Window to appear:



3. Now that you have the two windows displayed, you are ready to <u>assemble an application</u>.
   ♦ Click in the Network Editor title bar (above "File Edit Object") to bring the window to the front
   ♦ Look in the Visual Programming Area for the object "Uviewer".
   (You may need to enlarge the window by dragging the right hand lower corner down).
   ♦ Note the location on the "Uviewer" of the 2D Port and the 3D port as shown below:

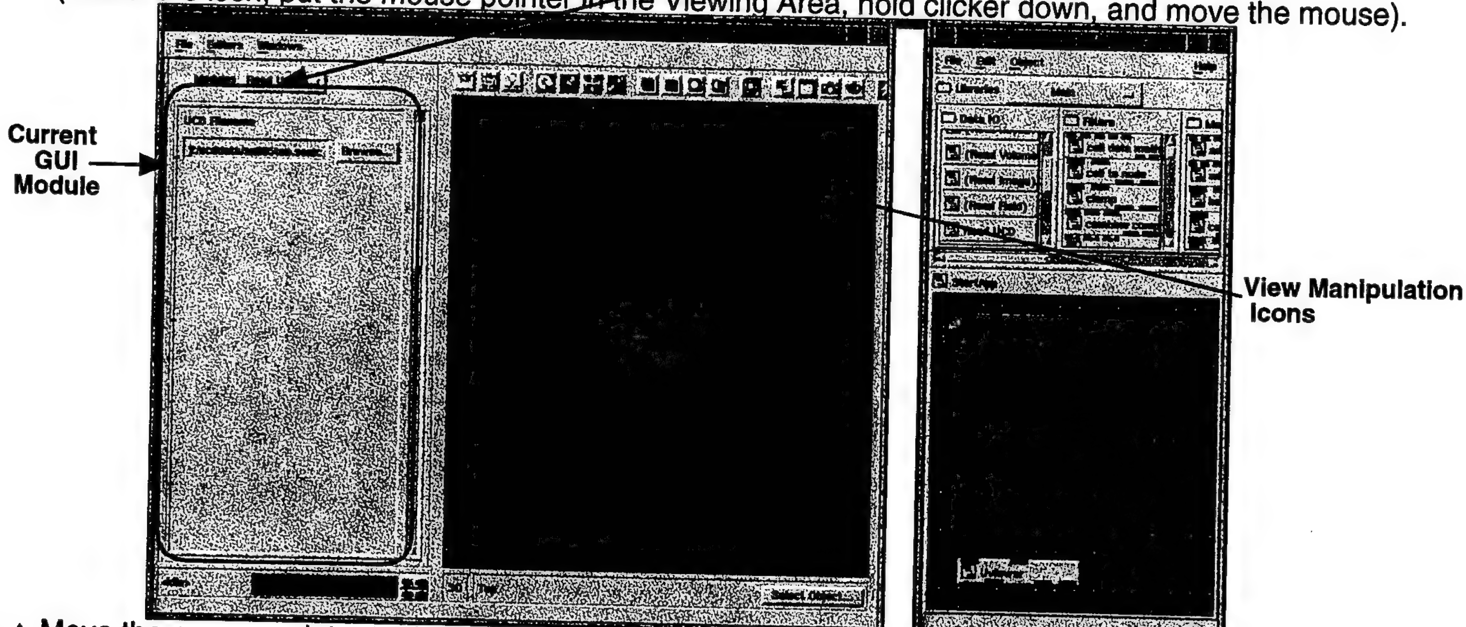


4. ♦ Drag the lower right hand corner of the Network Editor Window to the left to make it narrow (see below).
   ♦ Move the windows by dragging their title bars with the mouse pointer so you can see both at once.
   ♦ In Application Window, click on Graphic User Interface (GUI) "Modules" selection menu button
   ("Help" is highlighted when the clicker is down, because only the "Help" GUI module is available at present)



5. ♦ Move the mouse pointer to the Network Editor Window
   ♦ In the left most Library Folder (named "Data IO"), find the Visual Programming Object named "Read UCD" (Use the slider on the right of the folder contents to see all items, if necessary).
   ♦ Drag the "Read UCD" object from the "Data IO" library folder to the Visual Programming Area
   ♦ Connect the "Read UCD"'s red output port to the Uviewer's 3D input port (leftmost) by dragging the mouse pointer between the ports with the clicker down.

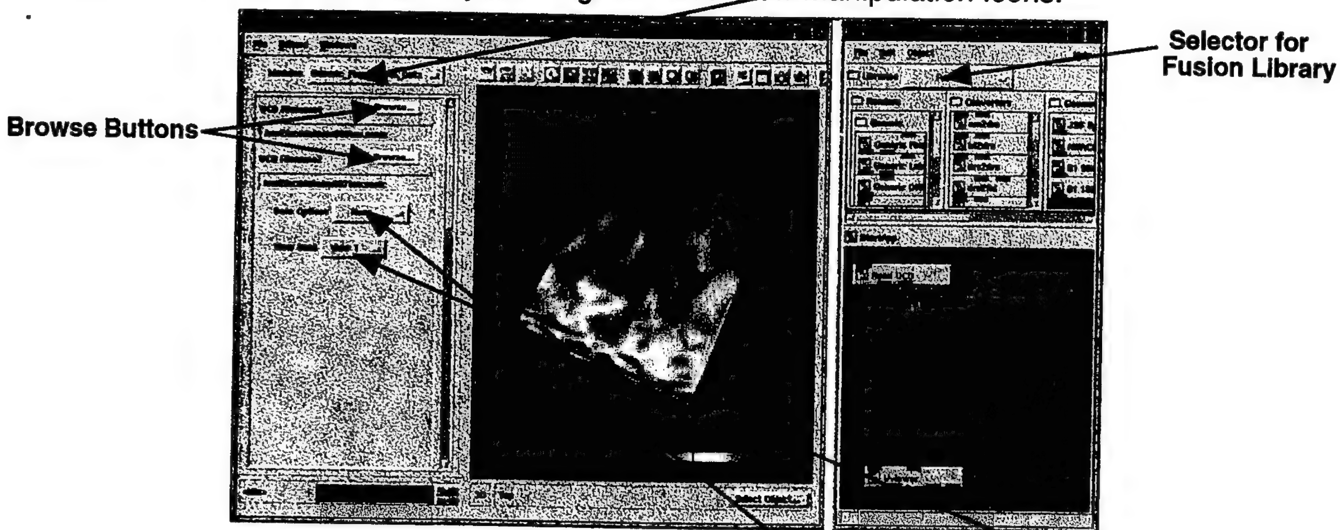
6. ♦ Move the mouse pointer to the Application Window
   ♦ Pull down the Graphic User Interface (GUI) "Modules" selection menu and select "Read UCD".
   ♦ Note the new Graphic User Interface module (GUI) which has appeared in the Application Window:
   ♦ Press the BROWSE button and enter (or browse to) file "/usr/people/john/.DATA/ueddata/sn005top.sonic". (Hint: browser starts at /usr/people/john/, so doubleclicking .DATA, then ueddata, in the list of Directories, then doubleclicking sn005top.sonic in the list of Files gets you to the file very quickly.)
   ♦ Click on the OK button before proceeding, if necessary. A data object should appear in the Viewing Area
   ♦ Rotate, move, and scale the object using the three View Manipulation Icons indicated below. (Select the icon, put the mouse pointer in the Viewing Area, hold clicker down, and move the mouse).



7. ♦ Move the mouse pointer to the Network Editor Window.
   ♦ Disconnect the Red Line connecting "Read UCD"s output port to "Uviewer"s 3D input port (Hint: Put mouse pointer on red line, push the right hand mouse button, and select "Delete Connection")

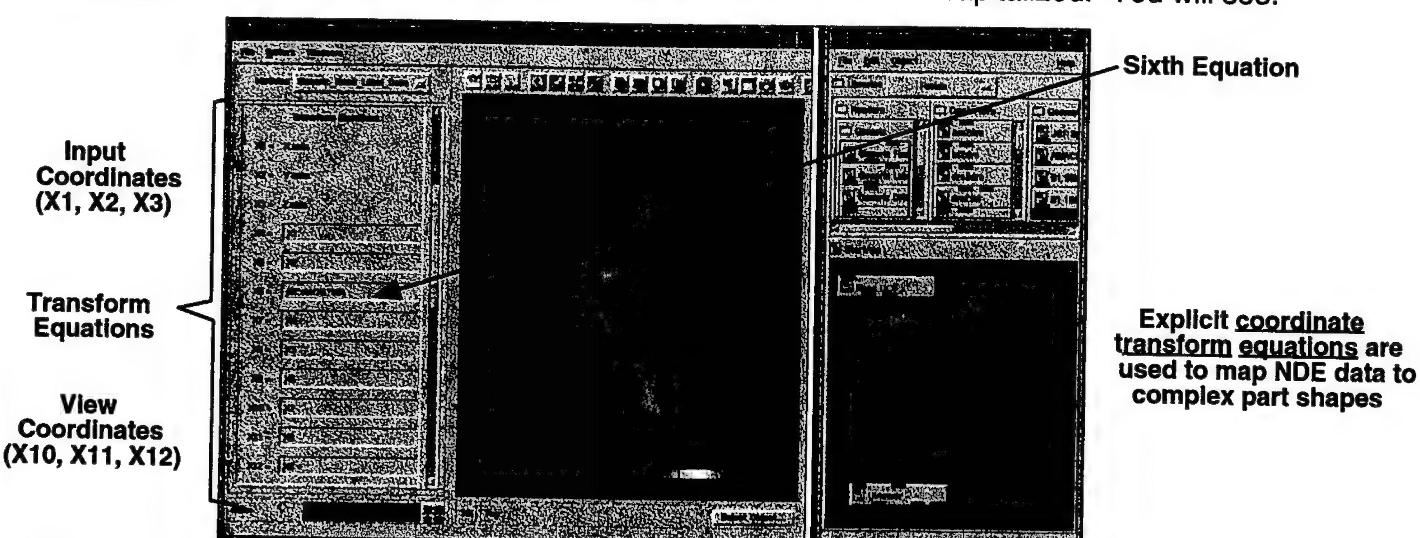
8. ♦ Locate the Fusion Library (next one under "Main")
   ♦ Get a "Generic Fixed Label Data" from the Reader folder in the Fusion Library, (hint: look in the "Generic" subfolder)
   ♦ Drag the "Generic Fixed Label Data" to the Visual Programming Area
   ♦ Connect the red ouput port of the "Generic Fixed Label Data" to the "Uviewer"'s 3D input port

9. ♦ Move the mouse pointer to the Application Window
   ♦ Pull down the Graphic User Interface (GUI) Selection Menu and select "Generic Fixed Label Data".
   ♦ Note the new Graphic User Interface module (GUI) which has appeared in the Application Window.
   ♦ Press upper yellow BROWSE button and browse to file "/usr/people/john/.DATA/ucddata/sn005top.sonic".
   ♦ Note the data object which appears in the Viewing Area
   ♦ Press lower yellow BROWSE button and browse to file "/usr/people/john/.DATA/ucddata/sn007top.sonic".
   ♦ Rotate, move, and scale the object using the three View Manipulation Icons.



10. The following steps may require some help from your trainer/mentor.
   ♦ First, try each of the four options provided by the "View Data" pull down menu. (Watch for changes in the Viewing Area when you select each option).
   ♦ Under the "Data Options" pull down menu (where it says "None"), select the "Features" option.
   ♦ Change the toggles for "data component". (Watch for changes in the Viewing Area when you select each option).

11. ♦ Change from "Features" to "Coordinates". Select "Defining Equations" from the "Select" pulldown menu
   ♦ Move the right hand slider bar in the the GUI area to view all 12 equations (The equations transform the UCD file's coordinates into viewer coordinates).
   ♦ If you are unfamiliar with the concept of coordinate transformations, ask your mentor to explain.
   ♦ Go to the 6th equation, click in the box showing "X3", type backspace twice, then type "40*sin(X1*10)" followed by <Enter> in the equation box . Be sure the "X" in "X1" is capitalized. You will see:



♦ Go back to the 6th equation, click in the box showing "40*sin(X1*10)" and change it back to "X3" (Watch the object in the Viewing Area as you make these changes)

12.♦ Move the mouse pointer back to the Network Editor Window.
♦ Disconnect the Red Line connecting "Generic Fixed Label Data"s output port to "Uviewer"s 3D input port
♦ Get a "Radome Top" from the Reader folder in the Fusion Library (Hint: look in the "AWACS" subfolder)
♦ Drag the "Radome Top" to the Visual Programming Area
♦ Connect the red ouput port of the "Radome Top" to the "Uviewer"'s 3D input port

13.♦ Move the mouse pointer to the Application Window and move the GUI slider to the top.
♦ Pull down the Graphic User Interface (GUI) Selection Menu and select "Radome Top". (Here)
♦ Press upper yellow BROWSE button and browse to file "/usr/people/john/.DATA/ucddata/sn005top.sonic".
♦ Be sure to click on the OK button before proceeding. A data object should appear in the Viewing Area
♦ Press lower yellow BROWSE button and browse to file "/usr/people/john/.DATA/ucddata/sn007top.sonic".
♦ Rotate and scale the object using theappropriateView Manipulation Icons.

14. ♦Repeat step 10 for this ("Radome Top") Graphic User Interface
♦Change from "Features" to "Coordinates". Select "Defining Equations" from the "Select" pulldown menu .
♦ Move the right hand slider bar in the the GUI area to view all 12 equations
.♦ Note that the only difference between a "Generic Fixed Label Data" and a "Radome Top" is that the latter has the equations to transform the data to the AWACS radome's surface geometry.
♦ Finally, get an "AWACS Radome" from the "Geometries" folder and connect it to the "Uviewer" as shown:

FINAL NOTE

You have used INDERS V3 to build a simple Data Fusion Application which used an unstructured cell data reader to build a transformable model of NDE data from ultrasonic testing of two AWACS radomes. In the subsequent exercises you will learn how to reuse the Application you have created, and how to convert raw NDE data into unstructured cell data.

4.2 Second Exercise – Assembling Ultrasonic Data Fusion Application

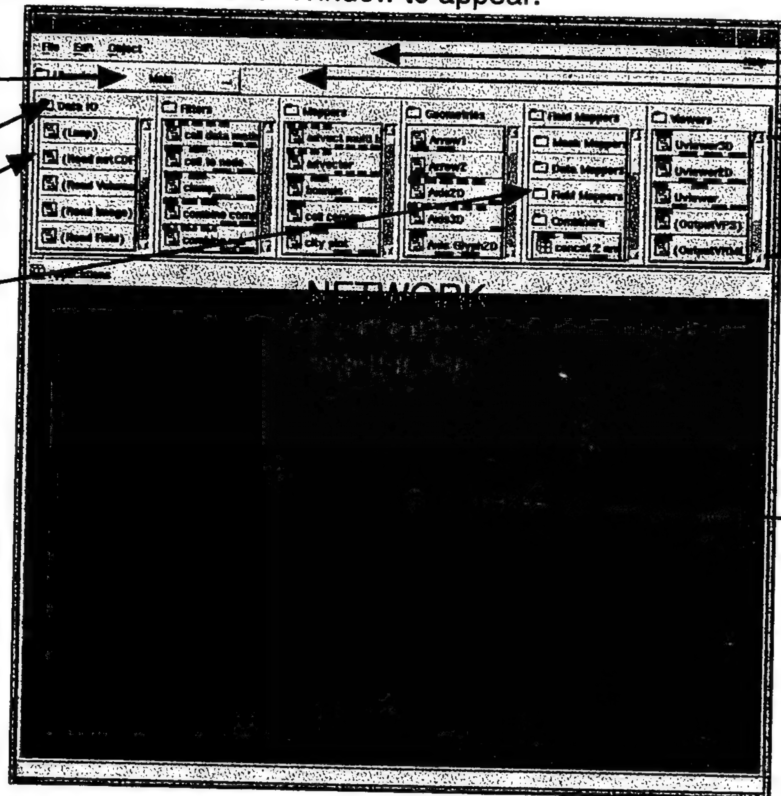
In this exercise, the user is introduced to some of the powerful signal processing and unstructured field conversion tools built into INDERS 3.

The *drs2fld* programming object combines an ultrasonic waveform viewer with a large set of digitial signal processing and signal feature extraction functions to produce a renderable (UCD) unstructured field object representing the ultrasonic data (i.e. generalized C–scan). Unstructured fields can be rendered as uniform, continu35
ous, or panelized (isolated sample) meshes and viewed using the objects illustrated in the first exercise. A complete set of processing options including analytic envelope generation, logarithmic decoding, coordinate or envelope amplitude based waveform selection, and autocorrelation features are provided.

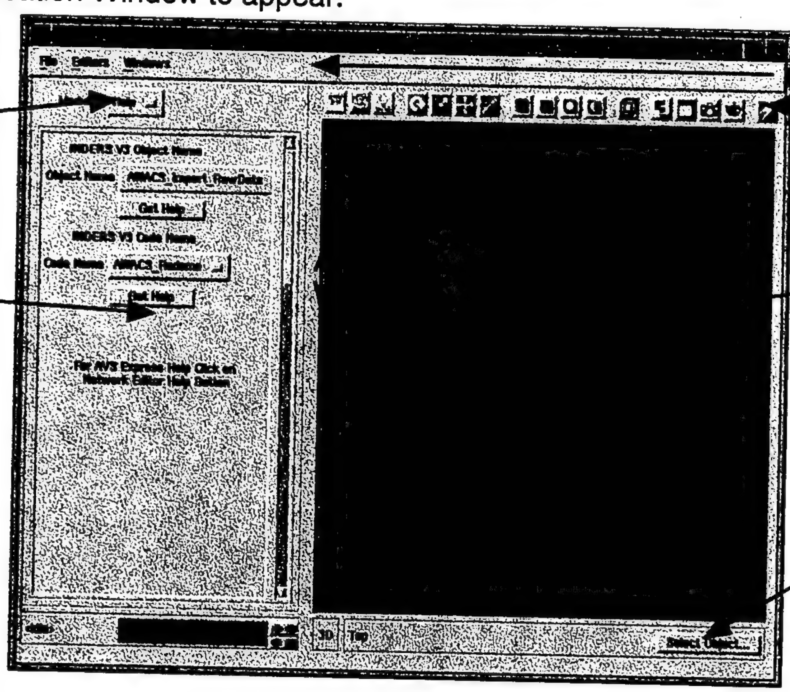
The exercise also introduces the interface for the invocation of external programs which is common to all of the objects in the *Converters* library and shows the INDERS 3 user how to save an application once he or she has assembled it.

ASSEMBLE ULTRASONIC (DRUS) DATA FUSION APPLICATION

1. ♦ Place the CD entitled "INDERS 3.1" into the CDROM holder, then into the CDROM Drive.
   ♦ Double–click (using the left mouse button) on the CDROM icon on your desktop.
   ♦ Double–click on the "xpr" icon in the CDROM window.
   ♦ Wait 2 minutes for the Network Editor Window to appear:

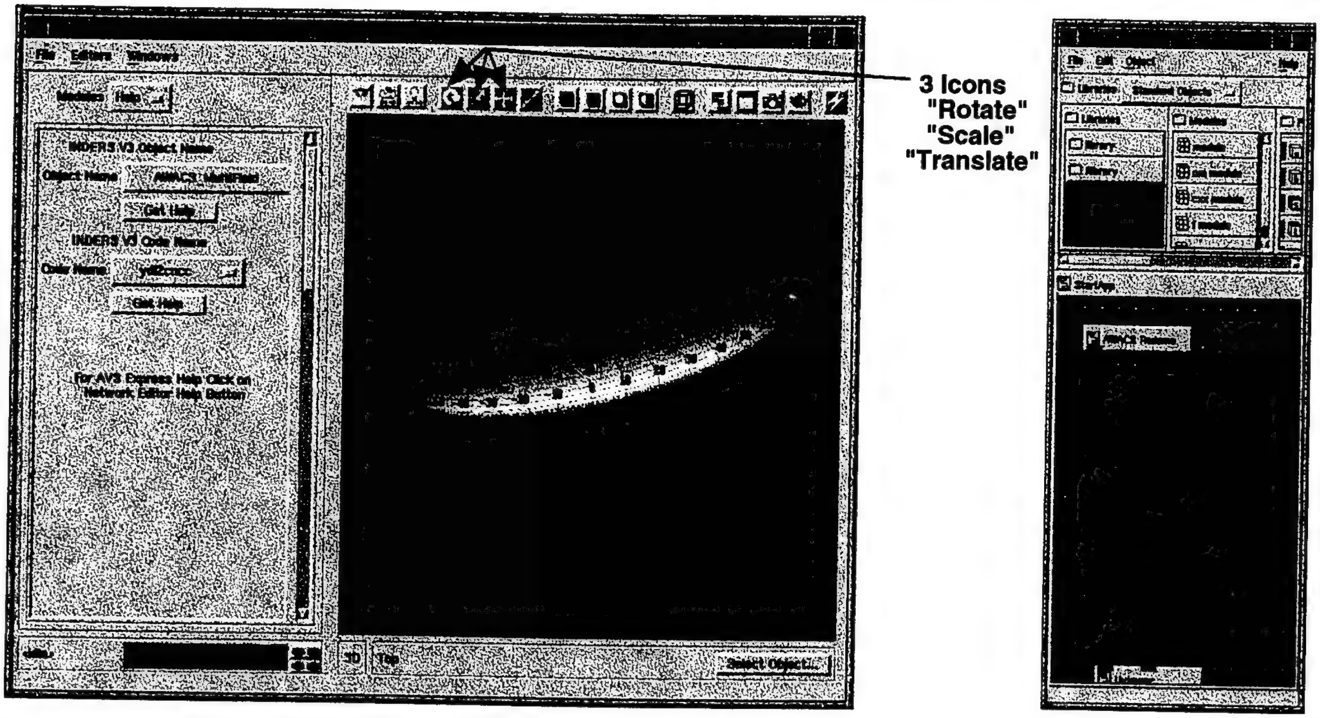


2. ♦ From the Main Network Editor Menu (top lefthand corner), select "File", then "New Application".
   ♦ Wait for the Application Window to appear:

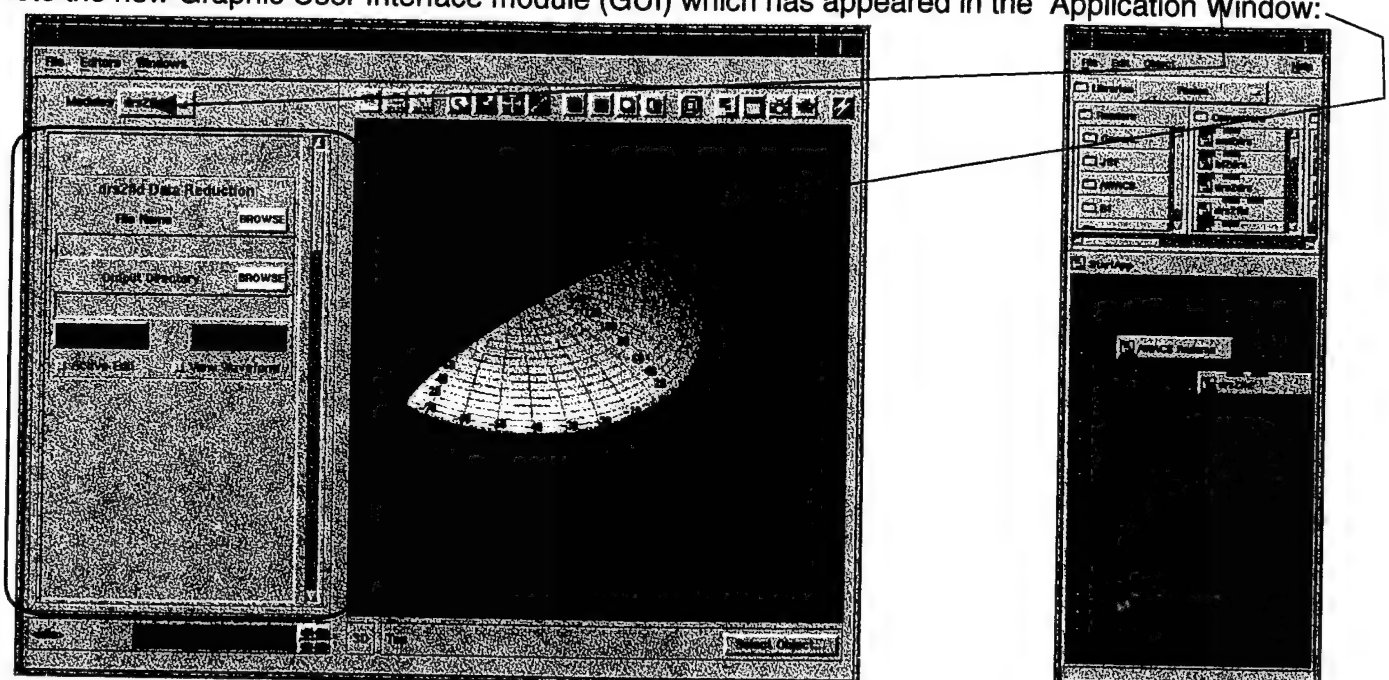


3. Now that you have the two windows displayed, you are ready to assemble an application.
   ♦ Click in the Network Editor title bar (above the Main Menu) to bring the window to the front
   ♦ Look in the Visual Programming Area for the object "Uviewer".
   (You may need to enlarge the window by dragging the right hand lower corner down).
   ♦ Note the location on the "Uviewer" of the 2D Port and the 3D port as shown below:

4. ♦ Drag the lower right hand corner of the Network Editor Window to the left to make it narrow (see below).
   ♦ Move the windows by dragging their title bars with the mouse pointer so you can see both at once.
   ♦ In the Network Editor, go to the pulldown menu under "Main" and select the Fusion Library.
   ♦ Drag the "AWACS Radome" object from Geometries library folder to the Visual Programming area.
   ♦ Connect the "AWACS Radome"'s red output port to the Uviewer's 3D input port (leftmost) by dragging the mouse pointer between the ports with the clicker down.
   ♦ Rotate, move, and scale the object using the three View Manipulation Icons indicated below. (Select the icon, put the mouse pointer in the Viewing Area, hold clicker down, and move the mouse).

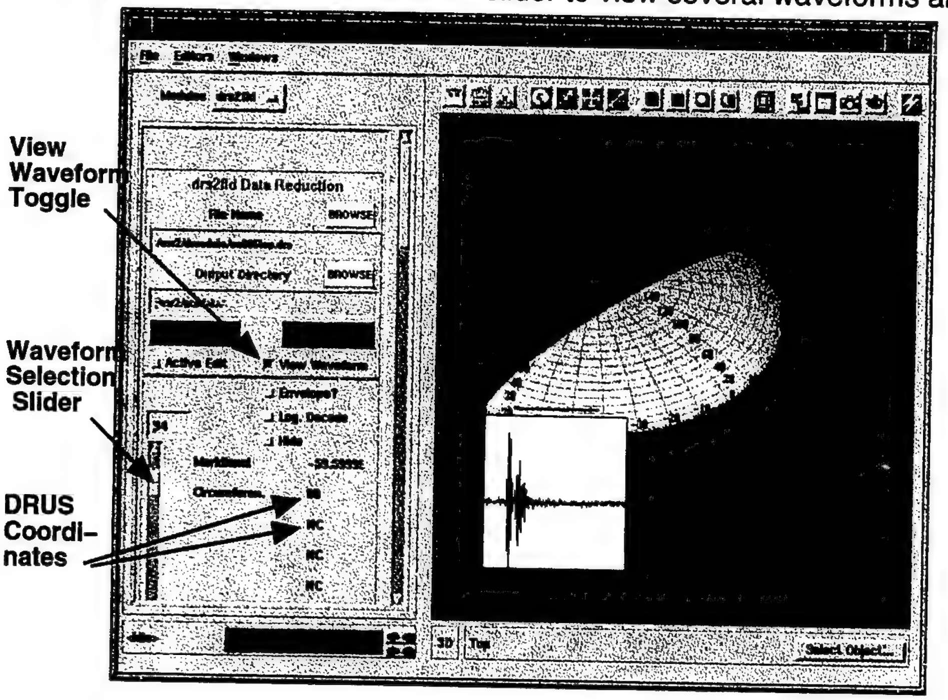


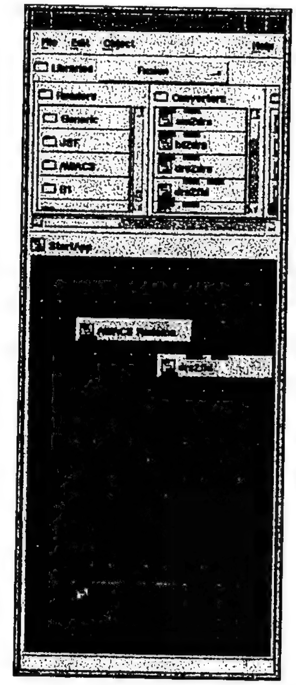
5. ♦ In the Network Editor, find the Converters folder for the "drs2fld" object.
   ♦ Drag it into the Visual Programming Area and connect it to the 2D port of Uviewer (as shown below).
   ♦ Pull down the Graphic User Interface (GUI) selection menu and select "drs2fld". (Here)
   ♦ Note the new Graphic User Interface module (GUI) which has appeared in the Application Window:



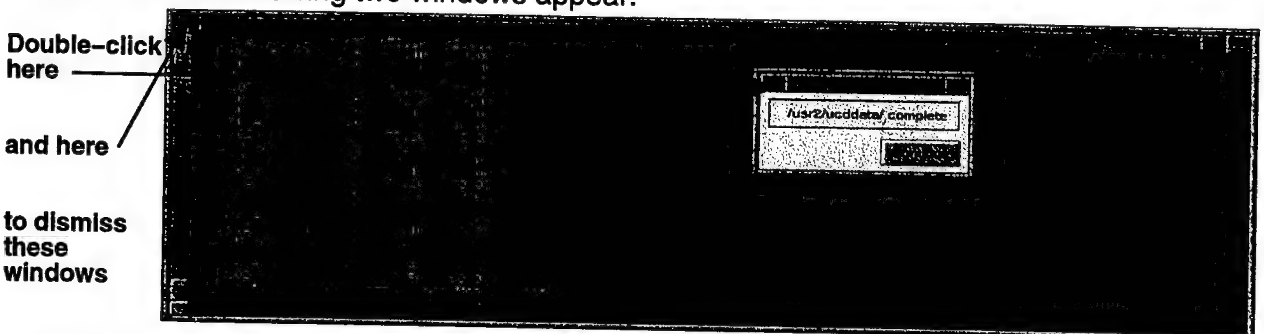
6. ♦ Press upper yellow BROWSE button and browse to <u>file</u> "/usr/people/john/.DATA/drusdata/sn005top.drs".
   ♦ Click on the OK button before proceeding, if necessary.
   ♦ Use the lower BROWSE button to select the output <u>file</u> "/usr/people/john/.DATA/ucddata/sn005top.ucd".
   ♦ (optional) Ultrasonic experts may wish to read about the ultrasonic signal analyzer "drs2fld" by: clicking on the blue "Help" button on the "drs2fld" GUI.
   (In step 8, use values of the parameters explained in the Help response rather than defaults.)

7. ♦ Click on the "View Waveform" toggle in the "drs2fld" GUI to graph the current waveform in Viewing Area. (You may have to wait for twenty or thirty seconds for the graph to appear).
♦ Move the waveform selection slider to view several waveforms and their DRUS coordinates(see below:



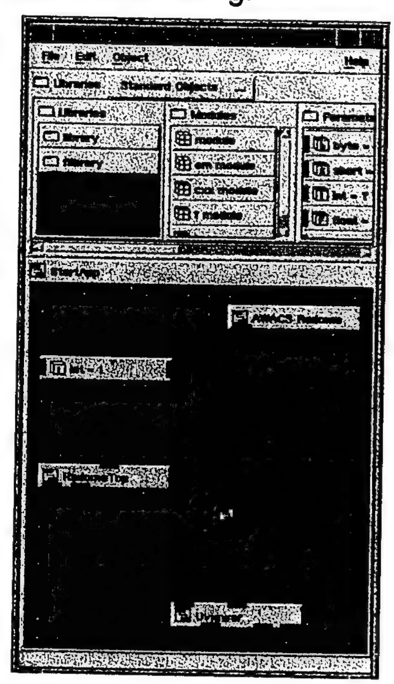


8. ♦Click on the "Active Edit" button and use the right hand slider to examine the ultrasonic signal feature extraction parameters available for modification in this GUI.
♦Click on the green "SUBMIT" button.
♦Wait until the following two windows appear:



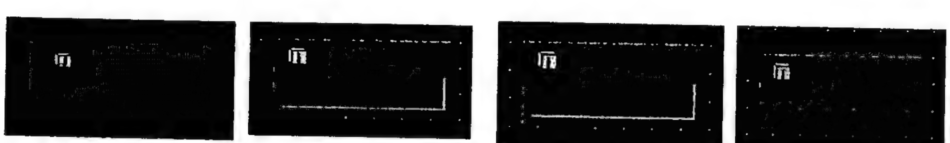
♦ These windows are for information only. You <u>must</u> dismiss them (as shown) before continuing.

9. Now that you are somewhat familiar with the Network Editor, see if you can add to your network as shown to the right. The steps are as follows:



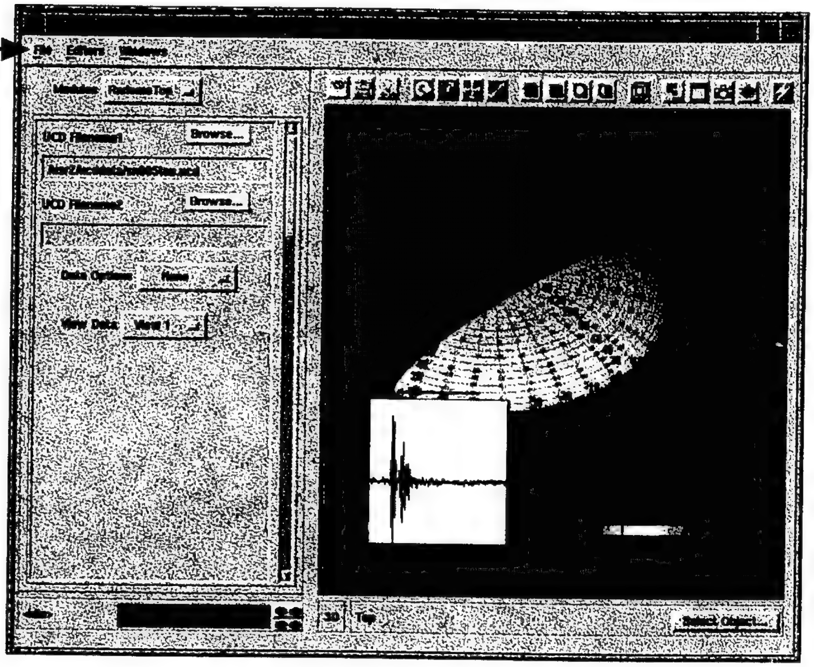
♦ get a "Radome Top" from the Fusion Library in the Reader folder, AWACS subfolder
♦ get an "int" from the Standard Objects Library in the Parameters folder (3rd from left, use slider on the right of the folder contents to see all items).
♦ set the "int" to a value of 1 by double clicking on it, to open typing "1" in the box, and double clicking on it again to close it (see below)
♦ connect the "int" to "Radome Top" and "Radome Top" to the "Uviewer"

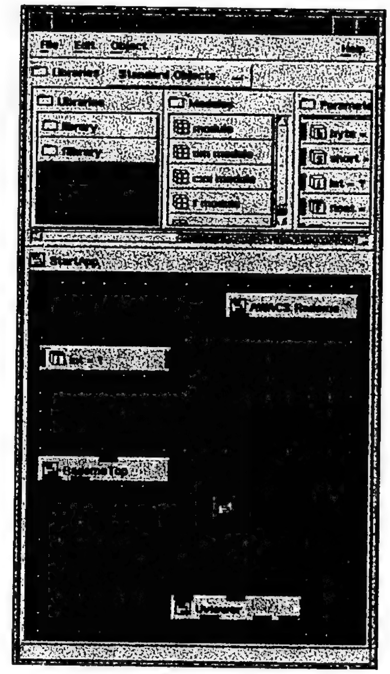
(Note: You may want to resize the Network Editor Window and rearrange the objects to make it easier to see the connections. Network Editor objects can be dragged and dropped with the mouse within the Visual Programming Area.)



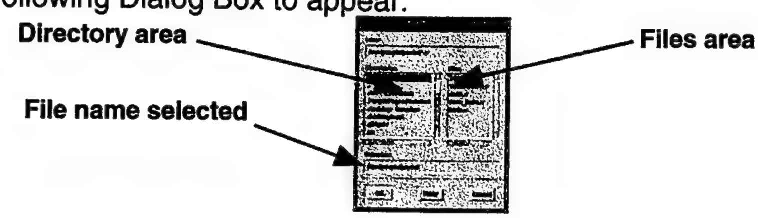
Double-click Type 1 then Enter Double-click again Result

10. ♦ Go to the Application Window and select the "Radome Top" GUI module.
- ♦ Use upper yellow BROWSE button to browse to the file " /usr/people/john/.DATA/ucddata/sn005top.ucd".
- ♦ Be sure to click on OK in the browser window before proceeding.
- ♦ Wait for the following display to appear in the Viewing Area:
- ♦ Try to change the value of "int" from 1 to 10. Note how the position of the labels and legend changes. (Mouse motions don't move labels and legends for "Radome Top" or other "Generic Fixed Label Data"s)





11. You are ready to save the INDERS V3 Application which you have just assembled.
- ♦ Go to the upper left of the Network Editor Window, select "File", and then "Save Application".
- ♦ Wait for the following Dialog Box to appear:



- ♦ Browse to the .APPLICATIONS directory and pick a filename identifying yourself (i.e. "JohnSmithApp.v") .
- ♦ Be sure to click on OK in the browser window.
- ♦ Exit AVS/Express by selecting "File", "Exit", from either window's Main Menu.

CONGRATULATIONS

You have used INDERS V3 to build a simple Data Fusion Application which combined geometry data for the AWACS radome with both full waveform and processed ultrasonic test data.

To reuse the Application you have created, repeat steps (1) and (2), then select "File", "Load Application", from the Network Editor Main Menu, browse to the Application you saved (i.e. "JohnSmithApp.v"), and hit OK.

Optional Recommended Further Steps Using the Application you created.

- ♦ examine the other options in the RadomeTop GUI including "Data Options", "Features", "Coordinates", and "View Data"
- ♦ change the toggles for "Step Colors", "Data Limits", "data component", etc. and see how the Visualization model changes.
- ♦ examine the options for the "drs2fld" GUI. Read the associated Help information. Note that feature code A6002 (AWACS autocorrelation features) and continuous (rather than panelized) mesh are more appropriate inputs to "drs2fld" than what you previously used.
- ♦ Try adding radar range test data and horn stethoscope (microwave reflectometer) data to the application. (Hint: the data objects needed are "Range Data" and "Horn Data" in the same folder where you found "Radome Top".)

5.0 Guide to INDERS 3 Visual Programming Objects

INDERS 3 uses AVS/Express as its programming platform. The organization of the existing AVS/Express object libraries has been preserved in INDERS 3, and an additional *Fusion* library has been added to augment the substatial AVS/Express functionality with specific tools designed for NDE data fusion. Because there are so many libraries and an hierarchy of folders and subfolders, the user may find it time consuming to locate the correct object or series of objects to perform a desired function. In addition, the programming objects are typically very generalized and multifunctional, so that using the name of the object alone to relate it to the desired functionality may not always be convenient.

For instance, during the development of INDERS 3, the author searched for a function to evaluate the data from one unstructured field on the mesh of another unstructured field (a common requirement of INDERS 1 and 2 analysis). After considerable investigation, it was discovered that the AVS/Express object *interp_data* performed this function, exactly as desired. However, because the AVS/Express developer's choice for the name of the object (which fit his perspective of its most important functionality), it had been overlooked by the author.

Note that the complete set of components contained in the nine AVS/Express libraries are fully documented in associated online AVS/Express reference manuals which are provided on the INDERS 3 CDROM. Reading the complete online reference manuals is the best way to become completely familiar with the existing functionality. However, the following guide should help a novice user get started by identifying commonly used components related to NDE data analysis and NDE data fusion.

5.1 Reading and Viewing Image Data

Common image data formats such as TIFF, BMP, etc. are read using the *Read_Image* object in the *Main* library. The visualizable output of the object (red port at the bottom of the programming object) can be connected directly either to the 2D or 3D port of the *Uviewer*. *Read_Image* will automatically determine the format of the image data file if the user doesn't select a particular format in the *Read_Image*'s GUI. It can read AVS X, BMP, GIF, JPEG, PBM, SGI Image, Sun Raster, and TIFF. The object *img2img* in the *Converters* folder in the *Fusion* library can be used to perform a file–to–file translation from many other image formats to one of these, so that virtually any image file format is accessible.

If NDE data fusion functionality is to be used, images must be rewritten to a UCD file using the object *DVwrite_ucd*, which is located in the *Primitives* folder in the *Visualization* library. The resultant UCD file will have explicit coordinates which can be transformed and manipulated using the *Generic_Fixed_Label_Data* or *Generic_Labelled_Data* objects in the *Readers* folder in the *Fusion* library.

If image processing functionality (such as FFT) is required, the result of *Read_Image* must be converted into IPimage format, then a series of objects from the Imaging library must be used. For users interested in this functionality, it is best to read the section <u>IP image processing macros</u> in the <u>Data Visualization Kit</u> online

reference manual.

5.2 Reading and Viewing Ultrasonic Data

Ultrasonic data is typically either full waveform or image data. Where the existing ultrasonic data is provided in C–scan image formats such as TIFF, refer to reading of image data in Section 5.1. For specialized or proprietary formats, tools in the Converters library are used to convert the raw ultrasonic data to either UCD (for B–scan or C–scan data) or DRUS (for full waveform ultrasonic data).

The object *maus2fld* converts MAUS ultrasound data directly to UCD where it can be manipulated using the *Fusion* library tools.

The object *am2drs* converts data from Panametrics (acoustic microscope) format to DRUS. Similarly, *bf2drs* converts data from Boeing's blade/fillet full waveform scanner's format to DRUS. The object *luis2drs* converts data from UltraOptek's LUIS format to DRUS. The object *sonix2drs* converts data from Sonix Corporation ultrasonic scanner format to DRUS.

Once the data is converted to DRUS, the object drs2fld in the Converters folder in the Fusion library is used both for waveform viewing, and to perform signal feature extraction and preparation of a calculated "C–scan". The resultant "C–scan" is a fully functional unstructured field, suitable for arbitrary data fusion operations. The object drs2fld is intended to encapsulate the complete ultrasonic signal processing functionality. It includes the ability for an advanced user to define new signal feature extraction methods by relinking with a user defined subroutine.

5.3 Reading CT Data and Converting to UCD

CT data is assumed to be in an INDERS 1 neutral file format. Stand–alone codes for performing conversions from AFACTS I and II, GE9800, GE9800/QUIC, GE XIM, SMS, ACTIS, ACTIS+, and IMATRON are provided on the CDROM. The object *neut2ucd* is used for converting multislice CT data into a renderable 3D visual object model suitable for NDE data fusion.

5.4 Building New Part Geometries

A number of objects are provided for building new part geometries.

The first recommended approach to translating from a CAD description of a part geometry is to generate a NASTRAN file consisting of triangle or quadrilateral elements and using the *nastran2ucd* object in the Converters library to convert the geometry into a UCD file. The object *step2ucd* may also be used to convert from ISO–10303–21 (STEP) geometry descriptions directly, but only if the STEP data contains the appropriate model. Most CAD systems are quite capable of generating a NASTRAN file, and this has proven to be a more robust pathway between CAD and INDERS 3 functionality than other approaches.

A second approach to building a part geometry is to manually assemble a UCD or V file describing the object or prepare a short computer program for writing the UCD description. This is appropriate for simple parts such as developmental hardware

(panels, T–sections, etc.) where the CAD geometry is unavailable.

The third method is CT reverse engineering of the part using the object *neut2ucd* in the *Converters* folder in the *Fusion* library. This is appropriate for small, complex parts which can be scanned on an approapriate CT system.

Generating the surface model is not necessarily sufficient for good NDE data fusion practice. Typically, an INDERS 3 part geometry consists of three elements, (a) the part itself, (b) chalklines, and (c) chalkline labels. Chalklines and chalkline labels represent part fixed coordinate axes. Unlike a space fixed set of coordinate axes they must be "chalked on" to the surface of the part.

In some cases the coordinate system selected is so simple (i.e. Cartesian) that an *Axis3D* object from the *Geometries* folder in the Main library can be used instead of explicit chalklines and labels. If it is attached to the geometry correctly, the system will remain fixed to the part, rather than the viewing space. A number of other tools are available in the *Geometries* folder in the *Main* library for building and labelling geometry objects. For instance, the object *textstring3D* is very useful for part fixed labelling of geometry objects.

5.5 Mapping UCD Data to Part Geometries

A unique feature of INDERS 3 is the ability to define an arbitrary, equation based nonlinear transformation to the NDE data. Since NDE data are converted to unstructured field descriptions (i.e. have explicit coordinates), these transformations do not operate on or compromise the data. From the user point of view, these transforms are defined in the GUIs for the objects in the *Readers* folder in the *Fusion* library such as *Generic_Labelled_Data* and *Generic_Fixed_Label_Data*.

A transform is typically between the coordinate system of the inspection system and the coordinate system of the part. Often, the coordinate system of the part is a simple 3D Cartesian system which reflects its position within an aircraft. This is particularly true for parts converted from CAD descriptions. In other cases, local coordinate systems are defined which relate to the symmetries and/or measurement fiducials which are present on the part.

The simplest method for mapping UCD data to the part geometry is to connect both the data (i.e. a *Generic_Fixed_Label_Data*) and the geometry to the 3D port of the *Uviewer* and to edit the equations in the GUI dynamically until the data is correctly mapped. Depending on the skill level of the user, this may be difficult to do "live", and may require offline algebraic manipulation. Any transformation equation defined in the object will be saved in the application, so it will not have to be re–entered. This is referred to as visual registration.

Alternatively, the object *Select_Point* in the *Miscellaneous* folder in the *Fusion* library allows users to move a pointer around the surface of both the data and geometry object in order to mark fiducials. The associated coordinates can then be used to derive a polynomial or other fit to the transformation equations. Currently, the fit is performed by using commercial programs such as Microsoft Excel or by implementing simple FORTRAN programs which call the INDERS library routine

SQRST for least squares solution of overdetermined systems of equations. The latter allows the user to specify an arbirary transform model, not just polynomial.

5.6 Other Commonly Used Programming Objects

The following is a list of visual programming objects which are most commonly used in current Boeing applications. Each is documented in associated online AVS/Express reference manuals which are provided on the INDERS 3 CDROM. The purpose of this table is to identify commonly used functions so the user can reduce the amount of hunting through all of the AVS/Express libraries and associated documentation.

Main Library
- Data IO Folder
  - Read_Image........................... Reads image files (ie. TIFF) into a uniform field [DV5.76]
  - Read_UCD............................. Reads UCD files into an unstructured field [DV5.79]
- Filters Folder
  - crop....................................... Crop data from edges of uniform field [DV5.18]
  - clamp.................................... Constrain data values to specified range [DV5.9]
  - data math............................... Arithmetic operations on matching fields [DV5.24]
  - extract scalar.......................... Extracts single data component from field [DV5.39]
- Mappers Folder
  - interp_data............................. Evaluates the data from one field onto the mesh of another field [DV5.48]
  - orthoslice................................ Extracts a single slice (ie. row or column) from a uniform field [DV5.63]
- Geometries
  - Axis3D.................................... Provides a complete set of object fixed coordinate axes including labels [DV6.5]
  - Cross3D.................................. A useful pointer geometry object (ie. used as part fixed mouse pointer) [DV6.10]
  - LegendHoriz........................... Horizontal data legend (ie. color scale) [DV6.20]
  - LegendVert............................ Vertical data legend [DV6.20]
  - TextString.............................. User specifiable text object for 2D. Used for titles and labelling of visualization data [GD6.62]
  - TextString3D......................... Same as TextString but moves & rotates 3D [GD6.62]
- Field Mappers Folder
  - Array Extractors Subfolder
    - extract_coordinates...... Extracts the coordinates of a field as a float array [DV5.36]

extract_data	Extracts the data values of a field as a float array [DV5.37]
Viewers Folder	
Uviewer	The INDERS standard viewer object. Supports both a 2D and 3D overlayed viewport [GD6.66]
Accessories Library	
Graphics Folder	
Viewer3D	Used for alternate 3D views in a separate window (without GUI panel or 2D overlay)
Utility Modules	
General Folder	
shell_command	Invokes a Unix or Windows shell command [DR7.15]
Field_Schema	
Primitives Folder	
DVswitch	Allows switching between a series of different visualizations [DV4.86]
DVinvert_xform	Used with Viewer3D to present "camera" view from within visualization space (ie. for radiographic view simulation)
Standard_Objects Library	
Parameters Folder	
int	Long integer number
float	Floating point numbber
string	Character string
boolean	Logical flag
enum	Variable which takes on limited set of values
Macros Folder	
macro	Container for programming objects. Very useful for keeping user networks from becoming crowded and hard to read
User_Interface Library	
Containers Folder	
UImod_panel	Container for GUI objects which connects the GUI to the left hand area of the Uviewer Window (the "Module" panel) [UI2.46]
Widgets Folder	
UIbutton	Push Button (stays down) [UI10.1]
UIdial	Dial [UI10.2]
UIfield	Box for entering data from keyboard [UI10.3]
UIlabel	Label for UIfield or other widgets [UI2.39]
UIslider	Slider [UI10.6]
UItoggle	Toggle [UI2.77]
Dialogs Folder	
UIdirectory_Dialog	Directory browser object [UI2.26]
UIfile_dialog	File browser object [UI2.26]
Annotation_Graphing Library	
Axes Folder	

AGXAxis...................... 2D graph x–axis definition [AG4.49]
AGYAxis...................... 2D graph y–axis definition [AG4.49]

Graphing Folder

AGGraph...................... 2D Graph [AG4.17]
AGGraphLegend.......... 2D Graph legend [AG4.19]
AGGraphWorld............ All above graph objects are inputs for this object [AG4.23]
AGGraphViewportObj. Converts the AGGraphWorld object into a displayable view, suitable for connecting directly to the Uviewer [AG4.22]

Imaging Library

Converters Folder

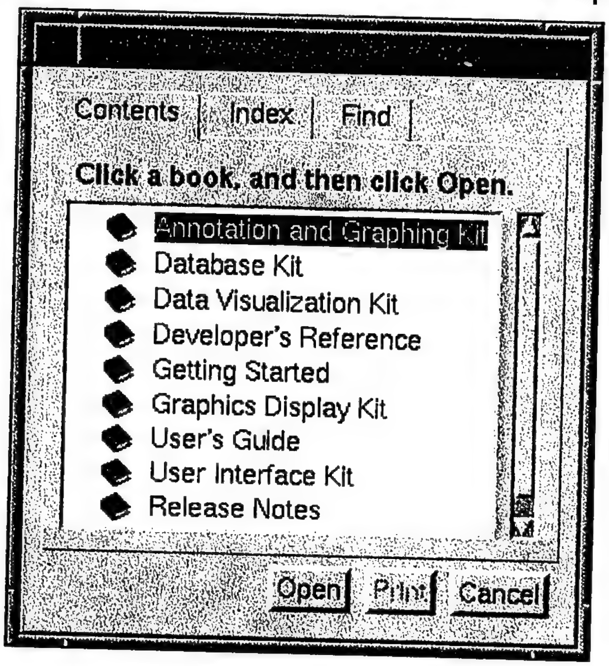
IPfldToImage............... Converts between uniform fields and Image Processable "image". (Not to be confused with image files like TIFF) [IP7.18]
IPimageToFld.............. Converts back [IP7.23]
IPfft.............................. Performs FFT on and IPimage [IP7.16]
IPfft_display................. Displays FFT on nonlinear scale for user [IP7.17]
IPifft............................. Performs inverse FFT on IPimage [IP7.22]
IPadd........................... Adds IPimages [IP7.3]
IPsubtract.................... Subtracts IPimages (multiply, divide, and numerous others are also available in the same folder) [IP7.X]
IPshift.......................... Shift an IPimage one pixel (ie. for shift and subtract operations) [IP7.42]

6.0 INDERS 3 Object Reference

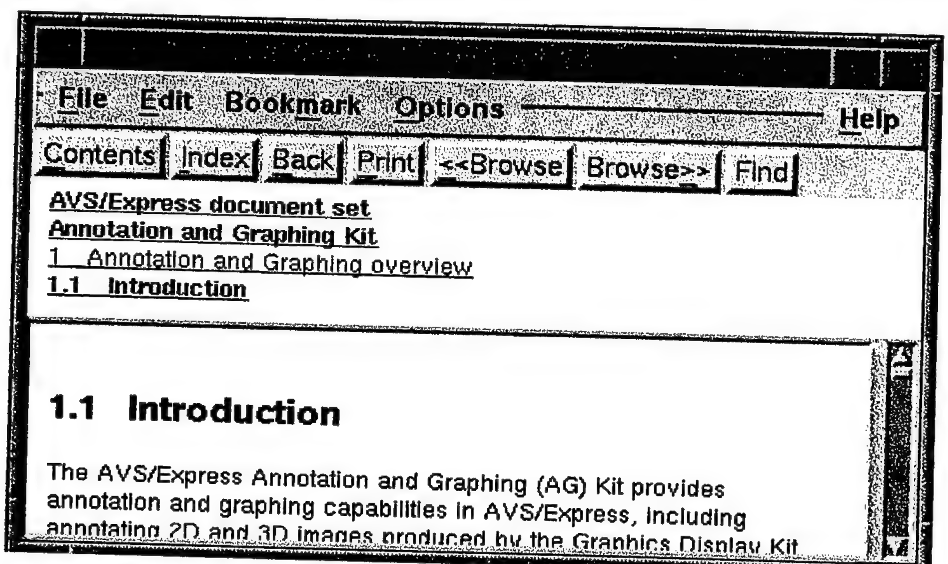
Rather than duplicate the voluminous online AVS/Express documentation here, instructions are provided for printing the documentation from an INDERS 3 CDROM.

6.1 Instructions for Printing AVS/Express Online Manuals

With INDERS 3 running on your workstation, click on the Help button in the upper right hand corner of the Network Editor Window, then the Contents pulldown selection. The AVS/Express Documentation window will appear on the screen.



Double–click on the topic desired, then on any subtopic until the following window appears.



Under the File pulldown menu, select Print, mark each of the Topics desired (use the shift key in conjunction with a left mouse click to mark additional Topics), and click on Ok. Choose Print Setup if you wish to reconfigure, print to a file, or select

other than the default printer.

6.2 Instructions for Printing AVS/Express Online Object Descriptions

With INDERS 3 running on your workstation, highlight the programming object of interest using the mouse. Then click on the Help button in the upper right hand corner of the Network Editor Window and select On Selected Object from the pull down menu. The AVS/Express Online Documentation Window will appear, with the reference description of the currently selected object showing.

Under the File pulldown menu, select Print and click on Ok in the Print Window. Choose Print Setup if you wish to reconfigure, print to a file, or select other than the default printer.

6.3 Fusion Library Object Descriptions

The following pages provide the the complete text of the online reference documentation for the programming objects contained in the *Fusion* library.

FUSION

This is the documentation set developed for use in INDERS data fusion.

1 Readers

Readers are modules that have been developed to read UCD data and apply user defined transforms to it. The Generic modules have a null transform, all others are specific to the geometry of the objects used by the programs in question.

1.1 Generic

The Generic Reader modules provide a standard interface for displaying UCD data in 3-D and DRUS Waveforms in 2-D. To create a new data application, use one of these modules as a starting point.

1.1.1 Generic Fixed Label Data

Synopsis
General INDERS3 data object with data labels fixed on the page.

Input Ports

Label_Location	int	specifies label location on page

Parameters

UCD Filename1	UIfileSB	name of the first file to read
UCD Filename2	UIfileSB	name of the second file to read
Data Option	UIoptionMenu	data manipulation GUI to display

View Data	UloptionMenu	view file 1, 2, difference or ratio

Data Option in Features Mode Only:

Data component	UlradioBox	pick which data feature to display
Step Colors	Ultoggle	if set, display <Num Steps> colors
Num Steps	Ulfield	number of display colors
Data Limits	Ultoggle	if set, set colors to data limits
Min	Ulfield	data min, Data Limit not set
Max	Ulfield	data max, Data Limit not set

Data Option in Coordinates Mode Only:
See Generic_MultiField

Output Ports

out_fld	Mesh+Cell_Data+Node_Data+Node_Data	port to pass data and coordinates
obj	DefaultObject	visualizable object

Description

Generic Fixed Label Data has a GUI that allows the user to select one or two files containing data to visualize. The files must be of the Application Visualization System UCD (.ucd) format . The GUI allows the user to choose whether to view on of the data files alone, or the difference or ratio of the data files. For viewing one file alone, the other file need not be specified.

The data is transformed according the coordinate transform equations specified in the Generic_MultiField. These equations can be modified by the user within the GUI. The GUI allows the user to select the component of the field data to be rendered. The data label, containing file name, date, component being rendered, data units, and a color scale will be fixed on the page and can not be rotated, scaled or transformed along with the data using the visualization tools. The label positioning and color can be modified using the Network Editor by modifying the Label_Position object within Generic_Fixed_Label_Data.

The data is rendered using a full spectrum of pseudo-colors. The GUI allows the user to modify this to a specific number of steps used for coloration. The user can also modify the limits of the scale used for coloration. This allows the user to set color ranges to correspond to data ranges.

Input Ports

Label_Location

A port to specify label location on the page. A zero places the label at the bottom of the page. Each increment will move the label up one position, so that several labels can be displayed on the same page by passing each instance of Generic_Fixed_Label_Data a different integer number.

Parameters

UCD Filename1

UlfileSB file browser. Selects the first disk file to input and display.

The input file is an Application Visualization System UCD (.ucd) format.

UCD Filename2

UlfileSB file browser. Selects the second disk file to input and display.

The input file is an Application Visualization System UCD (.ucd) format.

Data Option

UloptionMenu. Choose data option menu to display. Default is NONE. Features allows manipulation of display data and appearance. Coordinates allows manipulation of data mapping and units.

View Data

UloptionMenu. Choose to view data from UCD Filename1, UCD Filename2, UCD Filename1 – UCD Filename2 (difference), or UCD Filename1 / UCD Filename2 (ratio). The files must be of the same type (same number of features).

Data component

UlradioBox. Choose which data component to view. The feature names are retrieved from the UCD file and displayed here. Default is the first feature. Visible only when Data Option is in Features mode.

Step Colors

Ultoggle. Toggle between full color spectrum (256 colors) or a stepped color spectrum. Visible only when Data Option is in Features mode.

Num Steps

UIfield. Number of color steps when in Step Colors mode. Default is 8, maximum is 256. Active only when Step Colors is set. Visible only when Data Option is in Features mode.

Data Limits

UItoggle. Toggle between color scale based on Data Limits, or based on Min and Max specified by user. Visible only when Data Option is in Features mode.

Min

UIfield. Minimum value to use when setting color scale. Active only when Data Limits is not set. Visible only when Data Option is in Features mode.

Max

UIfield. Maximum value to use when setting color scale. Active only when Data Limits is not set. Visible only when Data Option is in Features mode.

Select and X1 to X12

See Generic_MultiField Visible only when Data Option is in Coordinates mode.

Output Ports

out_fld

Passes information on data and coordinates to modules such as WRITECSV.

obj

Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer port. Since data is planar, can be connected to a 2D port.

File

$TOPDIR/inders3.project/v/General.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Generic_Fixed_Label_Data.z

INDERS documentation set
Fusion
1 Readers
1.1 Generic
1.1.2 Generic Labelled Data

1.1.2 Generic Labelled Data

Synopsis

General INDERS3 data object with data labels attached to data object.

Input Ports

Label_Type	int	specifies label spacing

Parameters

UCD Filename1	UIfileSB	name of the first file to read
UCD Filename2	UIfileSB	name of the second file to read
Data Option	UIoptionMenu	data manipulation GUI to display
View Data	UIoptionMenu	view file 1, 2, difference or ratio

Data Option in Features Mode Only:

Data component	UIradioBox	pick which data feature to display
Step Colors	UItoggle	if set, display <Num Steps> colors
Num Steps	UIfield	number of display colors
Data Limits	UItoggle	if set, set colors to data limits
Min	UIfield	data min, Data Limit not set
Max	UIfield	data max, Data Limit not set

Data Option in Coordinates Mode Only:

See Generic_MultiField

Output Ports

out_fld	Mesh+Cell_Data+Node_Data+Node_Data	port to pass data and coordinates
obj	DefaultObject	visualizable object

Description

Generic Labelled Data has a GUI that allows the user to select one or two files containing data to visualize. The files must be of the Application Visualization System UCD (.ucd) format . The GUI allows the user to choose whether to view on of the data files alone, or the difference or ratio of the data files. For viewing one file alone, the other file need not be specified.

The data is transformed according the coordinate transform equations specified in the Generic_MultiField. These equations can be modified by the user within the GUI. The GUI allows the user to select the component of the field data to be rendered. The data label, containing file name, date, component being rendered, data units, and a color scale will be attached to the visualization and can be rotated, scaled or transformed along with the data using the visualization tools. The label positioning and color can be modified using the Network Editor by modifying the Label_Position object within Generic_Labelled_Data.

The data is rendered using a full spectrum of pseudo-colors. The GUI allows the user to modify this to a specific number of steps used for coloration. The user can also modify the limits of the scale used for coloration. This allows the user to set color ranges to correspond to data ranges.

Input Ports

Label_Type

A port to specify label spacing from data object. A zero is the default. See JSF_Data for more information on other options.

Parameters

UCD Filename1

UIfileSB file browser. Selects the first disk file to input and display.

The input file is an Application Visualization System UCD (.ucd) format.

UCD Filename2

UIfileSB file browser. Selects the second disk file to input and display..

The input file is an Application Visualization System UCD (.ucd) format.

Data Option

UIoptionMenu. Choose data option menu to display. Default is NONE. Features allows manipulation of display data and appearance. Coordinates allows manipulation of data mapping and units.

View Data

UIoptionMenu. Choose to view data from UCD Filename1, UCD Filename2, UCD Filename1 – UCD Filename2 (difference), or UCD Filename1 / UCD Filename2 (ratio). The files must be of the same type (same number of features).

Data component

UIradioBox. Choose which data component to view. The feature names are retrieved from the UCD file and displayed here. Default is the first feature. Visible only when Data Option is in Features mode.

Step Colors

UItoggle. Toggle between full color spectrum (256 colors) or a stepped color spectrum. Visible only when Data Option is in Features mode.

Num Steps

UIfield. Number of color steps when in Step Colors mode. Default is 8, maximum is 256. Active only when Step Colors is set. Visible only when Data Option is in Features mode.

Data Limits

UItoggle. Toggle between color scale based on Data Limits, or based on Min and Max specified by user. Visible only when Data Option is in Features mode.

Min

UIfield. Minimum value to use when setting color scale. Active only when Data Limits is not set. Visible only when Data Option is in Features mode.

Max

UIfield. Maximum value to use when setting color scale. Active only when Data Limits is not set. Visible only when Data Option is in Features mode.

Select and X1 to X12

See Generic_MultiField. Visible only when Data Option is in Coordinates mode.

Output Ports

out_fld

Passes information on data and coordinates to modules such as WRITECSV.

obj

Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer port. Since data is planar, can be connected to a 2D port.

File

$TOPDIR/inders3.project/v/General.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Generic_Labelled_Data.z

INDERS documentation set
Fusion
1 Readers
1.1 Generic
1.1.3 Generic DRUS Viewer

1.1.3 Generic DRUS Viewer

Synopsis

General INDERS3 DRUS data reader and plotter. Allow user to set up for and execute drs2fld.

Input Ports

None

Parameters

See drs2fld parameters

Output Ports

obj	DefaultObject	visualizable object

Description

Generic_DRUS_Viewer has a GUI that allows the user to select a file containing DRUS data to be analyzed. Once the file has been selected, the GUI shows the default values for execution of drs2fld. To change these values, toggle Active Edit.

To view a plot of the data, toggle View Waveform. This will show a plot of the current waveform. To move through the file, use the GUI slider to pick a waveform or type the waveform number into the display box. The coordinates of the waveform are displayed within the GUI. If the graph does not appear to be scaled correctly, select Rescale Graph. The waveform can be seen with or without an envelope. The waveform can also be viewed in log decoded from. To adjust the Hilbert Transform parameters, toggle Active Edit on and change n or cf.

To remove the plot from view, either toggle Hide or toggle View Waveform to the off position. The data, filename, parameter names and other information about the waveform are available by using the Network Editor to access the viewer_info object withing the Generic_DRUS_Viewer object.

When all the parameters needed by drs2fld have been filled in successfully, the Submit button is active. Pressing this button will spawn a separate window in which drs2fld is executed.

Parameters

See drs2fld parameters

Output Ports

obj

Provides the visualizable object to the Uviewer. Connect to a 2D port.

File

$TOPDIR/inders3.project/v/General.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Generic_DRUS_Viewer.z
INDERS documentation set
Fusion
1 Readers
1.1 Generic
1.1.4 Generic DRUS Reader

1.1.4 Generic DRUS Reader

Synopsis
General INDERS3 application used for reading DRUS data.

Input Ports

None

Parameters

File Name	UIfileSB	name of the file to read
Waveform to read	UIfield/UIslider	select waveform number
DRUS Coordinate Names	UIlabel	display DRUS coordinate names
DRUS Coordinate Values	UIlabel	display DRUS coordinate values

Output Ports

Data_Info	DefaultObject	visualizable object

Description

Generic_DRUS_Reader allows an easy interface to collect data from a DRUS file. There is no plot, as in Generic_DRUS_Viewer. A GUI is provided for selection of a specific waveform within the DRUS file.

Parameters

File Name
UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a EPRI DRUS (.drs) format.

Waveform to read
UIfield/UIslider. Waveform number may be selected either using slider or by typing in value. Slider is maximized at maximum number of waveforms.

DRUS Coordinate Names
UIlabel. Displays actual DRUS coordinate names, if available. If not available, uses X Axis, Y-Axis, Z-Axis.

DRUS Coordinate Values
UIlabel. Displays actual DRUS coordinate values for each waveform. If value = -999, NC is displayed.

Output Ports

Data_Info
Provides a group containing waveform information.

File
$TOPDIR/inders3.project/v/General.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Generic_DRUS_Reader.z
INDERS documentation set
Fusion
1 Readers
1.2 JSF

1.2 JSF

The JSF Reader modules provide JSF specific interface for displaying UCD data in 3-D and Blade/Fillet Waveforms in 2-D.

INDERS documentation set
Fusion
1 Readers
1.2 JSF
1.2.1 JSF Data

1.2.1 JSF Data

Synopsis

JSF specific INDERS3 data object with data labels attached to data object.

Input Ports

Label_Type	int	specifies label spacing

Parameters

See Generic Labelled Data object for information on parameters, except those in Coordinates. For Coordinates, see JSF MultiField

Output Ports

out_fld	Mesh+Cell_Data+Node_Data+Node_Data	port to pass data and coordinates
obj	DefaultObject	visualizable object

Description

JSF Data is created from the Generic Labelled Data object. It uses a JSF MultiField in place of the Generic MultiField.

Input Ports

Label_Type

A port to specify label spacing from data object. A zero indicates that label spacing should leave room for a JSF Spar object. One leaves less space, and two indicates that the JSF Spar object is missing and labels should be placed as close to the data as possible.

Parameters

See Generic Labelled Data object for information on parameters, except as detailed below.

X3 to X12

See JSF MultiField

Output Ports

out_fld

Passes information on data and coordinates to modules such as WRITECSV.

obj

Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer port. Since data is planar, can be connected to a 2D port.

File

$TOPDIR/inders3.project/v/JSF.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_JSF_Data.z

1.2.2 JSF Import RawData

Synopsis
JSF specific INDERS3 data reader and plotter. Allows interface to execute the codes bf2drus, drs2fld.

Input Ports

None

Parameters

Select Raw Data Type	UIoptionMenu	Select data type to convert

See help for data input modules for information on further parameters.

Output Ports

obj	DefaultObject	visualizable object

Description

JSF_Import_RawData has a GUI that allows the user to select the type of raw data to process. There are three choices: Blade/Fillet (BF), MAUS, or Thermocouple (TC). This selection will make visible the appropriate module, BF_Data_Selection, MAUS_Data_Selection, or TC_Data_Selection. See these modules for more information.

MAUS data has 1 toggle to turn "trimming" on or off. The "reduction factor" can be modified using the Network Editor. The default value is 4. Once a file has been selected, it will be processed using the INDERS function rdmasu. The output will be place in ../<filename>.maus.

TC data has two input fields, "minval" and "maxval". These fields allow the user to define the limits of the data spread. The "largest panel" and "reduction factor" can be modified using the Network Editor. Once a file has been selected, it will be processed using the INDERS function rdsylk. The output will be placed in ./filename.thermo.

Once the BF file has been selected, the GUI shows the default values for submission to the batch queue. To change the drs2fld values the user toggles "Active Edit". For more information on what these fields mean, see the man pages for bf2drus and drs2fld. To view a plot of the BF data the user toggles "View Waveform". This will show a plot of the current waveform. To move through the file, use the horizontal GUI slider to pick a waveform within a waveform file, or type the waveform number into the display box. Use the vertical slider to view another waveform file, or type in file number. The coordinates of the waveform are displayed within the GUI. If the graph does not appear to be scaled correctly, select "Rescale Graph". The waveform can be seen with or without an envelope. The waveform can also be viewed in log decoded form. To adjust the hilbert parameters, toggle "Active Edit" on and change n or cf.

To remove plot from view, either toggle "Hide" or toggle "View Waveform" to the off position. The data, filename, parameter names, and other information about the waveform are available by using the Network Editor to access the viewer_info object within the BF_Select object.

When all parameters needed by drs2fld have been filed in successfully, the "Submit to Queue" button is active. Pressing this button will send the command line to the file $HOME/Inders3_queue. If a DRUS file does not exist, bf2drus is added to the batch queue prior to drs2fld. To start the queue executing, choose "File" from the menu bar and select "Execute Batch Queue". The queue will execute and move the original command line to $HOME/Inders3_complete with a log in $HOME/Inders3_Log.

Parameters

Select Raw Data Type
UIoptionMenu. Allows the viewer to make visible either the BF_Data_Selection, MAUS_Data_Selection, or TC_Data_Selection modules. See help for each of these modules for information on further parameters.

Output Ports

obj
Provides the visualizable object to the Uviewer. Must be connected to a 2D port.

File
$TOPDIR/inders3.project/v/JSF.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_JSF_Import_RawData.z

INDERS documentation set
Fusion
1 Readers
1.3 AWACS

1.3 AWACS

The AWACS Reader modules provide an AWACS specific interface for displaying UCD data in 3-D and DRUS Waveforms in 2-D.

INDERS documentation set
Fusion
1 Readers
1.3 AWACS
1.3.1 RadomeTop

1.3.1 RadomeTop

Synopsis

AWACS specific INDERS3 data object with data labels fixed on the page.

Input Ports

Label_Location	int	specifies label location on the page

Parameters

See Generic Fixed Label Data object for information on parameters, except coordinates. See AWACS_tm_MultiField for coordinate information

Output Ports

out_fld	Mesh+Cell_Data+Node_Data+Node_Data	port to pass data and coordinates
obj	DefaultObject	visualizable object

Description

RadomeTop is created from the Generic Fixed Label Data object. It uses an AWACS_tm_MultiField in place of the Generic MultiField. The data appears mapped to the top surface of an AWACS radome. Use the AWACS_Radome object to visualize the radome. Used primarily to visualize sonic data.

Input Ports

Label_Location

A port to specify label location on the page. A zero places the label at the bottom of the page. Each increment will move the label up one position, so that several labels can be displayed on the same page by passing each instance a different integer number.

Parameters

See Generic Fixed Label Data object for information on parameters, except coordinates. See AWACS_tm_MultiField for coordinate information.

Output Ports

out_fld

Passes information on data and coordinates to modules such as WRITECSV.

obj

Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer port. Since data is planar, can be connected to a 2D port.

File
$TOPDIR/inders3.project/v/AWACS.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_RadomeTop.z

INDERS documentation set
Fusion
1 Readers
1.3 AWACS
1.3.2 RadomeBot

1.3.2 RadomeBot

Synopsis
AWACS specific INDERS3 data object with data labels fixed on the page.

Input Ports

Label_Location	int	specifies label location on the page

Parameters

See Generic Fixed_Label Data object for information on parameters, except coordinates. See AWACS_tm_MultiField for coordinate information.

Output Ports

out_fld	Mesh+Cell_Data+Node_Data+Node_Data	port to pass data and coordinates
obj	DefaultObject	visualizable object

Description

RadomeBot is created from the Generic Fixed Label Data object. It uses an AWACS_tm_MultiField in place of the Generic MultiField. The data appears mapped to the bottom surface of an AWACS radome. Use the AWACS_Radome object to visualize the radome. Used primarily to visualize sonic data.

Input Ports
Label_Location
A port to specify label location on the page. A zero places the label at the bottom of the page. Each increment will move the label up one position, so that several labels can be displayed on the same page by passing each instance a different integer number.

Parameters
See Generic Fixed_Label Data object for information on parameters, except coordinates. See AWACS_tm_MultiField for coordinate information.

Output Ports
out_fld
Passes information on data and coordinates to modules such as WRITECSV.

obj
Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer port. Since data is planar, can be connected to a 2D port.

File
$TOPDIR/inders3.project/v/AWACS.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_RadomeBot.z

1.3.4 Range Data

Synopsis
AWACS specific INDERS3 data object with data labels fixed on the page.

Input Ports

Label_Location	int	specifies label location on the page

Parameters

See Generic Fixed Label Data object for information on parameters, except coordinates. See AWACS tm MultiField for coordinate information.

Output Ports

out_fld	Mesh+Cell_Data+Node_Data+Node_Data	port to pass data and coordinates
obj	DefaultObject	visualizable object

Description

Range Data is created from the Generic Fixed Label Data object. It uses an AWACS tm MultiField in place of the Generic MultiField. The data appears mapped to the surface of an AWACS radome. Use the AWACS Radome object to visualize the radome. Used primarily for visualizing range data.

Input Ports
Label_Location
A port to specify label location on the page. A zero places the label at the bottom of the page. Each increment will move the label up one position, so that several labels can be displayed on the same page by passing each instance a different integer number.

Parameters
See Generic Fixed Label Data object for information on parameters, except coordinates. See AWACS tm MultiField for coordinate information.

Output Ports
out_fld
Passes information on data and coordinates to modules such as WRITECSV.

obj
Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer port. Since data is planar, can be connected to a 2D port.

File
$TOPDIR/inders3.project/v/AWACS.v

See also
-> $TOPDIR/inders/runtime/catman/a man/cat1/V3 Object Range Data.z

INDERS documentation set
Fusion
1 Readers
1.3 AWACS
1.3.4 Horn Data

1.3.4 Horn Data

Synopsis

AWACS specific INDERS3 data object with data labels fixed on the page. Spefically for displaying Horn Stethoscope data on a AWACS_Radome.

Input Ports

Label_Location	int	specifies label location on the page

Parameters

See Generic Fixed_Label Data object for information on parameters, except coordinates. See AWACS_e_MultiField for coordinate information.

Output Ports

out_fld	Mesh+Cell_Data+Node_Data+Node_Data	port to pass data and coordinates
obj	DefaultObject	visualizable object

Description

Horn Data is created from the Generic Fixed Label Data object. It uses an AWACS_e_MultiField in place of the Generic MultiField. The data appears mapped to the top surface of an AWACS radome. Use the AWACS_Radome object to visualize the radome. Used to visualize horn stethoscope data.

Input Ports

Label_Location

A port to specify label location on the page. A zero places the label at the bottom of the page. Each increment will move the label up one position, so that several labels can be displayed on the same page by passing each instance a different integer number.

Parameters

See Generic Fixed_Label Data object for information on parameters, except coordinates. See AWACS_e_MultiField for coordinate information.

Output Ports

out_fld

Passes information on data and coordinates to modules such as WRITECSV.

obj

Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer port. Since data is planar, can be connected to a 2D port.

File

$TOPDIR/inders3.project/v/AWACS.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Horn_Data.z

INDERS documentation set
Fusion
1 Readers
1.3 AWACS
1.3.5 AWACS Import RawData

1.3.5 AWACS Import RawData

Synopsis

AWACS specific INDERS3 data reader and plotter. Allows interface to execute the code drs2fld.

Input Ports

None

Parameters

Select Raw Data Type	UIoptionMenu	Select data type to convert

Output Ports

obj	DefaultObject	visualizable object

Description

AWACS_Import_RawData has a GUI that allows the user to select the type of DRUS data to process. This assumes that the appropriate INDERS external codes have been used to translate raw data into DRUS data. There are four choices: RadomeTop (sonix2drs converts to DRUS), RadomeBot (sonix2drs converts to DRUS), Range, and Horn (horn2drs converts to DRUS). This selection will make visible drs2fld object with the appropriate parameter settings for the data type.

Parameters

Select Raw Data Type

UIoptionMenu. Allows the viewer to make visible the drs2fld module with the appropriate parameter defaults for the data type.

Output Ports

obj

Provides the visualizable object to the Uviewer. Must be connected to a 2D port.

File

$TOPDIR/inders3.project/v/AWACS.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_AWACS_Import_RawData.z

INDERS documentation set
Fusion
1 Readers
1.4 B1

1.4 B1

The B1 Reader modules provide a B1 specific interface for displaying UCD data in 3-D and DRUS Waveforms in 2-D.

INDERS documentation set
Fusion
1 Readers
1.4 B1
1.4.1 AUSS_90in_Data

1.4.1 AUSS_90in_Data

Synopsis

B1 specific INDERS3 data object with data labels fixed on the page. Specifically for displaying AUSS data on a B1_90inDoor.

Input Ports

Label_Location	int	specifies label location on the page

Parameters

See Generic_Fixed_Label_Data object for information on parameters, except coordinates. See AUSS_90in_MultiField for coordinate information.

Output Ports

out_fld	Mesh+Cell_Data+Node_Data+Node_Data	port to pass data and coordinates
obj	DefaultObject	visualizable object

Description

AUSS_90in_Data is created from the Generic Fixed Label Data object. It uses an AUSS 90in MultiField in place of the Generic MultiField. The data appears mapped to the top surface of a B1 90in Door. Use the B1 90inDoor object to visualize the door. Used to visualize AUSS data.

Input Ports
Label_Location
A port to specify label location on the page. A zero places the label at the bottom of the page. Each increment will move the label up one position, so that several labels can be displayed on the same page by passing each instance a different integer number.

Parameters
See Generic Fixed Label Data object for information on parameters, except coordinates. See AUSS 90inDoor MultiField for coordinate information.

Output Ports
out_fld
Passes information on data and coordinates to modules such as WRITECSV.

obj
Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer port. Since data is planar, can be connected to a 2D port.

File
$TOPDIR/inders3.project/v/B1.v

See also
-> $TOPDIR/inders/runtime/catman/a man/cat1/V3 Object AUSS 90in Data.z

INDERS documentation set
Fusion
1 Readers
1.4 B1
1.4.2 AUSS 180in Data

1.4.2 AUSS_180in_Data

Synopsis
B1 specific INDERS3 data object with data labels fixed on the page. Specifically for displaying AUSS data on a B1 180inDoor.

Input Ports

Label_Location	int	specifies label location on the page

Parameters

See Generic Fixed Label Data object for information on parameters, except coordinates. See AUSS 180in MultiField for coordinate information.

Output Ports

out_fld	Mesh+Cell_Data+Node_Data+Node_Data	port to pass data and coordinates
obj	DefaultObject	visualizable object

Description

AUSS_180in_Data is created from the Generic Fixed Label Data object. It uses an AUSS 180in MultiField in place of the Generic MultiField. The data appears mapped to the top surface of a B1 180in Door. Use the B1 180inDoor object to visualize the door. Used to visualize AUSS data.

Input Ports
Label_Location
A port to specify label location on the page. A zero places the label at the bottom of the page. Each increment will move the label up one position, so that several labels can be displayed on the same page by passing each instance a different integer number.

Parameters

See Generic Fixed Label Data object for information on parameters, except coordinates. See AUSS 180inDoor MultiField for coordinate information.

Output Ports

out_fld

Passes information on data and coordinates to modules such as WRITECSV.

obj

Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer port. Since data is planar, can be connected to a 2D port.

File

$TOPDIR/inders3.project/v/B1.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_AUSS_180in_Data.z

INDERS documentation set
Fusion
1 Readers
1.5 IUS

1.5 IUS

The IUS Reader modules provide an IUS specific interface for displaying UCD data in 3-D and DRUS Waveforms in 2-D. Currently there are no modules in this section.

INDERS documentation set
Fusion
2 Converters

Converters are modules that contain GUI's to allow easy user interface between AVS/Express and external INDERS codes. These modules spawn separate windows for data processing.

2 Converters

INDERS3 document set
Fusion
2 Converters
2.1 am2drs

2.1 am2drs

Synopsis

convert ultrasonic waveform data from Panametrics Acoustic Microscope format to EPRI DRUS

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with am2drs man pages
Submit	UIbutton	spawn a window to execute am2drs
Active Edit	UItoggle	if set, allow edit of am2drs parameters
Filename	UItext	input filename as passed to am2drs

Output Ports

none

Description

GUI interface for am2drs external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a Panametrics Acoustic Microscope format (.scn) format.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .drs, which will be appended if it is omitted.

The output file is an EPRI DRUS (.drs) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for am2drs.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual am2drs code is executed. Output from am2drs will appear in this window. The Submit button is only active when all necessary parameters for am2drs have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

filename

UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

Interface to External Code

The external code am2drs is invoked using the shell script $HOME/INDERS_queue. The code am2drs reads from <filename.extension> and creates a DRUS file named: <filename>.drs, which is then copied to the desired output filename.

Example Contents of $HOME/INDERS_queue

am2drs /rawdata/gef32.scn
mv /rawdata/gef32.drs /drsdata/newname.drs
inform "/drsdata/newname.drs complete"

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/am2drs.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/am2drs.z

2.2 auss2ucd

Synopsis

convert ultrasonic waveform data from AUSS V or AUSS Advanced Database format to AVS UCD format

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with auss2ucd man pages
Submit	UIbutton	spawn a window to execute auss2ucd
Active Edit	UItoggle	if set, allow edit of auss2ucd parameters
trim	UItoggle	trim zero data in image buffer
downsampling factor	UIradioBox	downsampling factor when trim is set
Filename	UItext	input filename as passed to auss2ucd
switches	UItext	displays the selected switches

Output Ports

none

Description

GUI interface for auss2ucd external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is an AUSS V or AUSS Advanced Database format.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .drs, which will be appended if it is omitted.

The output file is an AVS UCD (.ucd) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for auss2ucd.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual auss2ucd code is executed. Output from auss2ucd will appear in this window. The Submit button is only active when all necessary parameters for auss2ucd have been supplied.

Active Edit
UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

trim
UItoggle. Trim wasted space (zero data) in image buffer.

downsampling factor
UIradioBox. Chose downsampling factor when trim is set. Default is 1 when trim is not set.

filename
UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

switches
UItext. Displays the selected switches. When this field is blank, there will be no trimming or downsampling. This should not be changed directly. To set trimming or downsampling please use the trim UItoggle and the downsampling factor UIradioBox.

Interface to External Code
The external code auss2ucd is invoked using the shell script $HOME/INDERS_queue. The code auss2ucd reads from <filename.extension> and creates an AVS UCD file named: <filename>.ucd, which is then copied to the desired output filename.

Example Contents of $HOME/INDERS_queue
auss2ucd /aussdata/B100
mv /aussdata/B100.ucd /ucddata/newname.ucd
inform "/ucddata/newname.ucd complete"

or with trimming and downsampling:

auss2ucd /aussdata/B100 t5
mv /aussdata/B100.ucd /ucddata/newname.ucd
inform "/ucddata/newname.ucd complete"

File
$TOPDIR/inders3.project/v/converters.v

See also
-> $TOPDIR/inders3.project/src/inders3.codes/auss2ucd.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/auass2ucd.z

INDERS3 document set
Fusion
2 Converters
2.3 bf2drs

2.3 bf2drs

Synopsis
convert ultrasonic waveform data from blade/fillet UT scanner format to EPRI DRUS

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

Directory	UIdirectorySB	name of the directory with files to be read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with bf2drus man pages
Submit	UIbutton	spawn a window to execute bf2drus
Active Edit	UItoggle	if set, allow edit of bf2drus parameters
directory	UItext	input directory as passed to bf2drus
switches	UItext	switches as passed to bf2drus

Output Ports

none

Description

GUI interface for bf2drus external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

Directory

UIdirectorySB directory browser. Selects the disk directory that contains the set of disk files to input and convert. The default search pattern is $Input_Dir/*.

The input files are in blade/fillet UT scanner format. Directory selected must contain a file named *.SCN.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .drs, which will be appended if it is omitted.

The output file is an EPRI DRUS (.drs) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for bf2drs.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual bf2drus code is executed. Output from bf2drs will appear in this window. The Submit button is only active when all necessary parameters for bf2drs have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

D (delete raw files)

UItoggle. Activating this toggle forces bf2drus to delete the raw files after processing. May be changed only be changed in Active Edit mode.

directory

UItext. Displays the selected input directory. This is for confirmation only. Any changes in input directory should be made using the Directory browser.

switches

UItext. Displays the selected switches. The o switch is automatically added to force output to the desired directory. This should not be changed directly. The only other switch active at this time is the d (delete raw files) switch, which should be activated/deactived using the UItoggle.

Interface to External Code

The external code bf2drus is invoked using the shell script $HOME/INDERS_queue. The code bf2drus reads a series of ultrasonic B-scan signal data files that are stored in the specified directory. A DRUS file is created with the same name as the directory and extension of .DRS, which is then copied to the desired output filename.

Example Contents of $HOME/INDERS_queue

bf2drus /rawdata/971010 o/drsdata
mv /drsdata/971010.DRS /drsdata/newname.drs
inform "/drsdata/newname.drs complete"

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/bf2drus.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/bf2drus.z

2.4 drs2drs

Synopsis

selectively edit a DRUS file to eliminate repeated waveforms (ie. waveforms whose first two DRUS coordinates are coincident

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UlfileSB	name of the file to read
Output File	UlfileSB	name of the file to write
Help	Ulbutton	spawn a window with dr2drlast man pages
Submit	Ulbutton	spawn a window to execute dr2drlast
Active Edit	Ultoggle	if set, allow edit of dr2drlast parameters
Delete Repeated Points	UlradioBox	selects which waveform will be kept
Tolerance=<value>	Ulfield	determines which waveforms are the same
Transform DRUS Coordinates	UlradioBox	selects which coordinate to transform
Linear Coeff=<value>	Ulfield	linear coefficient of the transformation
Constant Coeff=<value>	Ulfield	constant coefficient of the transformation
Filename	Ultext	input filename as passed to dr2drlast
Tolerance	Ultext	tolerance as passed to dr2drlast
Transforms	Ultext	coordinate transform as passed to dr2drlast

Output Ports

none

Description

GUI interface for dr2drlast external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UlfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a EPRI DRUS (.drs) format.

Output File

UlfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .drs, which will be appended if it is omitted.

The output file is an EPRI DRUS (.drs) format.

Help

Ulbutton. Pressing this button causes a new window to be spawned containing the man pages for dr2drlast.

Submit

Ulbutton. Pressing this button causes a new window to be spawned, in which the actual dr2drlast code is executed. Output from dr2drlast will appear in this window. The Submit button is only active when all necessary parameters for dr2drlast have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

Delete Repeated Points

UIradioBox. Radio buttons to pick whether to keep only the First or Last waveform in the input file, where there are repeated waveforms for a given coordinate pair. Default is Keep Last. May only be changed in Active Edit mode.

Tolerance=<value>

UIfield. Tolerance is used to determine which waveforms are the same. The default value is 0.00001. The value displayed in the label is the value that will be passed to dr2drlast. May only be changed in Active Edit mode.

Transform DRUS Coordinates

UIradioBox. Selects which DRUS coordinate to transform linearly. The choices are None (default), First DRUS Coord, or Second DRUS Coord. Currently, only the first two DRUS coordinates can be transformed. May only be changed in Active Edit mode.

Linear Coeff=<value>

UIfield. If the user has selected a DRUS coordinate to transform, this value becomes the linear coefficient of the transformation. The value displayed in the label is the value that will be passed to dr2drlast. May only be changed in Active Edit mode.

Constant Coeff=<value>

UIfield. If the user has selected a DRUS coordinate to transform, this value becomes the constant coefficient of the transformation. The value displayed in the label is the value that will be passed to dr2drlast. May only be changed in Active Edit mode.

filename

UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

Tolerance

UItext. Displays tolerance as passed to dr2drlast. If Keep First is chosen, Tolerance should be negative. If Keep Last is chosen, Tolerance should be positive. This is for confirmation only. Any changes should be made using the Delete Repeated Points UIradioBox and Tolerance UIfield.

Transforms

UItext. Displays the DRUS coordinate transformation as passed to dr2drlast. If NONE was chosen, this will be blank. If first DRUS Coord is chosen, the display will contain 1,<Linear Coeff>,<Const Coeff>. If second DRUS Coord is chosen, the display will contain 1,<Linear Coeff>,<Const Coeff>.. This is for confirmation only. Any changes should be made using the Transform DRUS Coordinates UIradioBox, Linear Coeff UIfield, and Const Coeff UIfield.

Interface to External Code

The external code dr2drlast is invoked using the shell script $HOME/INDERS_queue. The code dr2drlast reads from <filename.drs> and writes a file into the current directory named TEMP.DRS, which is then copied to the desired output filename. If tol is omitted it is assumed to be 0.00001. If tol is a positive number, only the last waveform in the DRUS file for a given coordinate pair is retained in the output file. If tol is a negative number, only the first waveform in the DRUS file for a given coordinate pair is retained in the output file.

If the third argument appears (requires a tol), the coordinates of the output DRUS file are transformed linearly from the input coordinates. For instance, an argument of

    1,-1.0,0.0

would reverse the sign of the 1st DRUS coordinate. Currently, only the first two DRUS coordinates are edittable in this way.

Example Contents of $HOME/INDERS_queue

```
dr2drlast /rawdata/bigfile.drs 0.05 1,-1.0,0.0
mv /rawdata/TEMP.DRS /drsdata/newname.drs
inform "/drsdata/newname.drs complete"
```

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/dr2drlast.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/dr2drlast.z

INDERS3 document set
Fusion
2 Converters
2.5 drs2fld

2.5 drs2fld

Synopsis

General INDERS3 feature extraction program. Reads EPRI DRUS (.drs) format file and creates visualizable 3D object which is the result of the feature extraction and object definition parameters supplied. The output file is an Application Visualization System UCD (.ucd) format.

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with drs2fld man pages
Submit	UIbutton	spawn a window to execute drs2fld
Active Edit	UItoggle	if set, allow edit of drs2fld parameters
View Waveform	UItoggle	if set, display DRUS waveforms
dt=<value>	UIfield	time sample interval
ts=<value>	UIfield	feature extraction window start time
te=<value>	UIfield	feature extraction window end time
tsm=<value>	UIfield	mean evaluation window start time
tem=<value>	UIfield	mean evaluation window end time
xmin=<value>	UIfield	lower range of first DRUS coordinate
xmax=<value>	UIfield	upper range of first DRUS coordinate.
ymin=<value>	UIfield	lower range of second DRUS coordinate
ymax=<value>	UIfield	upper range of second DRUS coordinate
zmin=<value>	UIfield	lower range of third DRUS coordinate
zmax=<value>	UIfield	upper range of third DRUS coordinate
amin=<value>	UIfield	lower range of signal amplitude
amax=<value>	UIfield	upper range of signal amplitude
Number of Gates	UIfield	number of gates used in extraction
Start Time for Gate n	UIfield	start time of the nth gate
End Time for Gate n	UIfield	end time of the nth gate
n=<value>	UIfield	halfwidth of Hilbert Transform convolution filter
cf=<value>	UIfield	Hilbert Tranform correction factor
irf=<value>	UIfield	selects type of mesh object to create
px=<value>	UIfield	x size of the discontinuous panel mesh
py=<value>	UIfield	y size of the discontinuous panel mesh
tolavg=<value>	UIfield	tolerance for determining coincidence
Code (A600nn)	UItext	selects the feature extraction process
Filename	UItext	input filename as passed to drs2fld
sigwins	UItext	signal window parameters as passed to drs2fld
exclbounds	UItext	exclusion bounds parameters as passed to drs2fld
gates	UItext	gate parameters as passed to drs2fld
htparams	UItext	Hilbert Transform parameters as passed to drs2fld
objparams	UItext	mesh object parameters as passed to drs2fld
featurecode	UItext	feature code parameter as passed to drs2fld

View Waveform Mode Only:

Type of waveform display	UIoptionBox	control waveform type
Waveform to display	UIfield/UIslider	select waveform number
DRUS Coordinate Names	UIlabel	display DRUS coordinate names
DRUS Coordinate Values	UIlabel	display DRUS coordinate values
Rescale Graph	UIbutton	activates automatic scaling

Output Ports

obj	DefaultObject	output renderable object

Description

GUI interface for drs2fld external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed. Also allows user to view the individual DRUS waveforms.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a EPRI DRUS (.drs) format.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .ucd, which will be appended if it is omitted.

The output file is an Application Visualization System UCD (.ucd) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for drs2fld.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual drs2fld code is executed. Output from drs2fld will appear in this window. The Submit button is only active when all necessary parameters for drs2fld have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

View Waveform

UItoggle. Activating this toggle brings up an additional GUI to control DRUS waveform viewing. Also allows visibility of the DRUS waveforms in the 2D viewer if obj is connected.

Type of waveform display

UIoptionBox. Option buttons to add to waveform display a Log decode waveform, or to display Envelope rather than raw data. The waveform display can also be hidden using the Hide button. If nothing is selected, the raw waveform will be displayed. Visible only in View Waveform mode.

Waveform to display

UIfield/UIslider. Waveform number may be selected either using slider or by typing in value. Slider is maximized at maximum number of waveforms. Visible only in View Waveform mode.

DRUS Coordinate Names

UIlabel. Displays actual DRUS coordinate names, if available. If not available, uses X Axis, Y-Axis, Z-Axis. Visible only in View Waveform mode.

DRUS Coordinate Values

UIlabel. Displays actual DRUS coordinate values for each waveform. If value = -999, NC is displayed. Visible only in View Waveform mode.

Rescale Graph

UIbutton. Allows the user to activate automatic scaling. Visible only in View Waveform mode.

dt=<value>

UIfield. Time sample interval in microseconds as read from input file. For information only. Never active.

ts=<value>

UIfield. Time feature extraction window starts in microseconds. Default is zero. May only be changed in Active Edit mode.

te=<value>

UIfield. Time feature extraction window ends in microseconds. Default is maximum time in input file. May only be changed in Active Edit mode.

tsm=<value>

UIfield. Time mean evaluation window starts in microseconds. Default is zero. May only be changed in Active Edit mode.

tem=<value>

UIfield. Time mean evaluation window ends in microseconds. Default is zero. May only be changed in Active Edit mode.

xmin=<value>

UIfield. Admissible lower range for the first DRUS coordinate. Default is –999 (no lower limit). May only be changed in Active Edit mode.

xmax=<value>
Ulfield. Admissible upper range for the first DRUS coordinate. Default is 999 (no upper limit). May only be changed in Active Edit mode.

ymin=<value>
Ulfield. Admissible lower range for the second DRUS coordinate. Default is –999 (no lower limit). May only be changed in Active Edit mode.

ymax=<value>
Ulfield. Admissible upper range for the second DRUS coordinate. Default is 999 (no upper limit). May only be changed in Active Edit mode.

zmin=<value>
Ulfield. Admissible lower range for the third DRUS coordinate. Default is –999 (no lower limit). May only be changed in Active Edit mode.

zmax=<value>
Ulfield. Admissible upper range for the third DRUS coordinate. Default is 999 (no upper limit). May only be changed in Active Edit mode.

amin=<value>
Ulfield. Admissible lower range for the signal amplitude. Default is –1e6 (no lower limit). May only be changed in Active Edit mode.

amax=<value>
Ulfield. Admissible upper range for the signal amplitude. Default is 1e6 (no upper limit). May only be changed in Active Edit mode.

Number of Gates
Ulfield. Minimum of 1 gate and maximum of 10 gates. An input line will be visible for each gate requested. Default is 1. May only be changed in Active Edit mode.

Start Time for Gate n
Ulfield. Start time of the nth gate in microseconds. Default is 0.0. May only be changed in Active Edit mode.

End Time for Gate n
Ulfield. End time of the nth gate in microseconds. Default is 0.0. May only be changed in Active Edit mode.

n=<value>
Ulfield. Halfwidth of the Hilbert Transform convolution filter. Default is 8. Must be a power of 2. May only be changed in Active Edit mode.

cf=<value>
Ulfield. Hilbert Tranform correction factor. Default is 1.17. If sampling frequency is much greater than the signal's central frequency, this number should be modified. May only be changed in Active Edit mode.

irf=<value>
Ulfield. If positive, irf is the reduction factor for a uniform mesh object (uniform in the coordinate space defined by the first two DRUS coordinates. If negative, irf selects a discontinuous panel mesh. If zero, irf selects a continuous triangulation mesh. Default is –1 (discontinuous panel mesh). May only be changed in Active Edit mode.

px=<value>
Ulfield. Inactive if irf is positive. If irf is negative, provides the first DRUS coordinate dimension size of the discontinuous panel mesh. If irf is zero, this value is ignored but must be present. Default is 1.0. May only be changed in Active Edit mode.

py=<value>
Ulfield. Inactive if irf is positive. If irf is negative, provides the second DRUS coordinate dimension size of the discontinuous panel mesh. If irf is zero, this value is ignored but must be present. Default is 3.0. May only be changed in Active Edit mode.

tolavg=<value>
Ulfield. Inactive if irf is positive. If irf is negative or zero, forces drs2fld to combine (average) nodal values for nodes which are within tolavg of being coincident (in the coordinate space defined by the first two DRUS coordinates). Default is 0.0 (must be exactly coincident). May only be changed in Active Edit mode.

Code (A600nn)
Ultext. Selects the feature extraction process . Currently defined options are:

A6000 - default gated envelope peak amplitudes and times
A6001 - JSF Weldline features, uses only 1st three gates
A6002 - AWACS Paint thickness features, uses only one gate
A6003 - Range data feature extraction, thresholds of response subtending 6 degrees, angle at peak response
A6004 - AWACS Horn Stethoscope Standard feature extraction, currently calculates gated peak amplitudes only

A6005 - titanium panel thickness UT Standard feature extraction for LUIS times and thicknesses by leading edge, autocorrelation of envelope, like A6002, but with pre-highpass filtering (because of LUIS)
A6006 - LUIS pulse to pulse and part to part variability features
A6007 - Guided mode interference between reflected and radiated burst features
A6022 - General autocorrelation feature extraction, uses 2 gates. Gates 1 and 2 bound the portion of the record processed in the autocorrelation calculation. Gates 3 and 4 bound the offset used in the calculation.

No checking is done to determine validity of the entry. Default is A6000. May only be changed in Active Edit mode.

filename
Ultext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

sigwins
Ultext. Displays the signal window parameters as passed to drs2fld. Of the format <dt>,<ts>,<te>,<tsm>,<tem>. This is for confirmation only. Any changes should be made using the dt, ts, te, tsm, and tem Ulfields.

exclbounds
Ultext. Displays the exclusion bounds parameters as passed to drs2fld. Of the format <xmin>,<xmax>,<ymin>,<ymax>,<zmin>,<zmax>,<amin>,<amax>. This is for confirmation only. Any changes should be made using the xmin, xmax, ymin, ymax, zmin, zmax, amin, and amax Ulfields.

gates
Ultext. Displays the gate parameters as passed to drs2fld. Of the format <Number of Gates>,<Start Time for Gate 1>,<End Time for Gate 1>,<Start Time for Gate 2>,<End Time for Gate 2>,...,<Start Time for Gate n>,<End Time for Gate n>. This is for confirmation only. Any changes should be made using the Number of Gates, Start Time for Gate n and End Time for Gate n Ulfields.

htparams
Ultext. Displays the Hilbert Transform parameters as passed to drs2fld. Of the format <n>,<cf>. This is for confirmation only. Any changes should be made using the n and cf Ulfields.

objparams
Ultext. Displays the mesh object parameters as passed to drs2fld. Of the format <irf>,<px>,<py>,<tolavg> if irf is negative or zero. Of the format <irf> if irf is positive. This is for confirmation only. Any changes should be made using the irf, px, py, and tolavg Ulfields.

featurecode
Ultext. Displays the feature code parameter as passed to drs2fld. This is for confirmation only. Any changes should be made using the Code (A60nn) Ulfield.

Output Ports
obj
This is the graph of the waveform and must be connected to the 2D port of the UViewer to be visible.

Interface to External Code
The external code drs2fld is invoked using the shell script $HOME/INDERS_queue. drs2fld reads from <filename> assumed to be in DRUS format.

<sigwins> is a comma separated 5-tuple of numbers: dt,ts,te,tsm,tem
where dt = time sample interval in microseconds
ts = time feature extraction window starts in microseconds
te = time feature extraction window ends in microseconds
tsm = time mean evaluation window starts in microseconds
tem = time mean evaluation window ends in microseconds

<exclbounds> is either a comma separated 8-tuple or a comma separated 14-tuple of numbers. If it is an 8-tuple it takes the form: xmin,xmax,ymin,ymax,zmin,zmax,amin,amax
where xmin and xmax define the admissible range for the 1st DRUS coordinate and ymin and xmax define the admissible range for the 2nd DRUS coordinate and zmin and zmax define the admissible range for the 3rd DRUS coordinate and amin and amax define the admissible range of signal amplitudes

If it is a 14-tuple, ranges for all six DRUS coordinates are provided rather than just the first three. The effect is to not perform feature extraction (in other words ignore) signals outside the specified ranges.

<gates> is a (2n+1)-tuple of comma separated numbers: n,ts1,te1,ts2,te2,...tsn,ten
where n is the number of gates
ts1 is the start time of the 1st gate
te1 is the end time of the 1st gate
ts2 is the start time of the 2nd gate
..etc.

<htparams> is the Hilbert Transform parameters, a comma separated 2-tuple of numbers: n,cf
where n = halfwidth of the convolution filter cf = the corresponding correction factor usually defaulted to 8,1.17 but different if the sampling frequency is much greater than the signal's central frequency.

<objparams> is either a 1-tuple or a 4-tuple of comma separated numbers: irf or irf,px,py,tolavg
where if irf is positive (1-tuple case) it is the reduction factor for a uniform mesh object (uniform in the coordinate space defined by 1st two DRUS coordinates) or if irf is negative, it selects a discontinuous panel mesh whose panels have size px and py in the 1st two DRUS coordinate dimensions, and which will combine (average) panel values for panels which are within tolavg of being coincident (in the coordinate space defined by 1st two DRUS coordinates) or if irf is zero, it selects a continuous triangulated mesh which will combine (average) nodal values for nodes which are within tolavg of being coincident (in the coordinate space defined by 1st two DRUS coordinates). In the latter case, px and py are not used but must appear in the 4-tuple.

<featurecode> is a string of the form A6nnn which denotes the feature extraction process selected. For each feature extraction process, there is a corresponding subroutine, and they can be added to by a process described here (but not yet written). See above for currently defined options.

Example Contents of $HOME/INDERS_queue

drs2fld /drsdata/drsfile.drs 0.01,0,81.9,0,.5 –999,999,-999,999,-999,999,-1e6,1e6 1,0.1,1.6 8,1.17 –1,1,3,0 A6000
mv /drsdata/drsfile.ucd /ucddata/newname.ucd
inform "/ucddata/newname.ucd complete"

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/drs2fld.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/drs2fld.z

INDERS3 document set
Fusion
2 Converters
2.6 drs2v

2.6 drs2v

Synopsis

convert ultrasonic waveform data from EPRI DRUS format to AVS/Express V language

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with drus2v man pages
Submit	UIbutton	spawn a window to execute drus2v
Active Edit	UItoggle	if set, allow edit of drus2v parameters
filename	UItext	input filename as passed to drus2v

Output Ports

none

Description

GUI interface for drus2v external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UlfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is an EPRI DRUS (.drs) format.

Output File

UlfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .v, which will be appended if it is omitted.

The output file is an AVS/Express V language (.v) format.

Help

Ulbutton. Pressing this button causes a new window to be spawned containing the man pages for drus2v.

Submit

Ulbutton. Pressing this button causes a new window to be spawned, in which the actual drus2v code is executed. Output from drus2v will appear in this window. The Submit button is only active when all necessary parameters for drus2v have been supplied.

Active Edit

Ultoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

filename

Ultext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

Interface to External Code

The external code drus2v is invoked using the shell script $HOME/INDERS_queue. The code drus2v reads a DRUS file and writes V files hedr.v and data.v. hedr.v describes the ultrasonic test information while data.v describes an AVS/Express object hierarchy containing the DRUS waveforms and their analytic envelopes. hedr.v and data.v are copied sequentially into the desired output filename.

Example Contents of $HOME/INDERS_queue

```
drus2v /drsdata/drsfile.drs
cat hedr.v data.v > /vdata/newname.v
inform "/vdata/newname.v complete"
```

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/drus2v.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/drus2v.z

INDERS3 document set
Fusion
2 Converters
2.7 horn2drs

2.7 horn2drs

Synopsis

convert horn stethoscope data (reflectance vs. frequency) to EPRI DRUS

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

Directory	UldirectorySB	name of the directory with files to be read
Output File	UlfileSB	name of the file to write
Help	Ulbutton	spawn window with horn2drus man pages
Submit	Ulbutton	spawn a window to execute horn2drus

Active Edit	UItoggle	if set, allow edit of horn2drus parameters
directory	UItext	input directory as passed to horn2drus

Output Ports

none

Description

GUI interface for horn2drus external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

Directory

UIdirectorySB directory browser. Selects the disk directory that contains the set of disk files to input and convert. The default search pattern is $Input_Dir/*.

The input files are in horn stethoscope data format (reflectance vs. frequency) in RTF files.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .drs, which will be appended if it is omitted.

The output file is an EPRI DRUS (.drs) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for horn2drs.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual horn2drus code is executed. Output from horn2drs will appear in this window. The Submit button is only active when all necessary parameters for horn2drs have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

directory

UItext. Displays the selected input directory. This is for confirmation only. Any changes in input directory should be made using the Directory browser.

Interface to External Code

The external code horn2drus is invoked using the shell script $HOME/INDERS_queue. The code horn2drus reads files from the directory and writes a DRUS file named <directoryname>.DRS, which is then copied to the desired output filename.

Example Contents of $HOME/INDERS_queue

```
horn2drus /rawdata/971010
mv /drsdata/971010.DRS /drsdata/newname.drs
inform "/drsdata/newname.drs complete"
```

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/horn2drus.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/horn2drus.z

2.8 hrrtr2fld

Synopsis
convert SunVision format to AVS/Express field format

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with sv2fld man pages
Submit	UIbutton	spawn a window to execute sv2fld
Active Edit	UItoggle	if set, allow edit of sv2fld parameters
Filename	UItext	input filename as passed to sv2fld

Output Ports

none

Description

GUI interface for sv2fld external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports
Input_Dir
A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory
A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters
File Name
UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a SunVision format.

Output File
UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .fld, which will be appended if it is omitted.

The output file is an AVS field (.fld) format.

Help
UIbutton. Pressing this button causes a new window to be spawned containing the man pages for sv2fld.

Submit
UIbutton. Pressing this button causes a new window to be spawned, in which the actual sv2fld code is executed. Output from sv2fld will appear in this window. The Submit button is only active when all necessary parameters for sv2fld have been supplied.

Active Edit
UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

filename
UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

Interface to External Code
The external code sv2fld is invoked using the shell script $HOME/INDERS_queue. The code sv2fld reads from <filename.extension> and creates an AVS field file named: <filename>.fld, which is then copied to the desired output filename.

Example Contents of $HOME/INDERS_queue
sv2fld /hrrtrdata/abc
mv /hrrtrdata/abc.fld /flddata/newname.fld
inform "/hrrtrdata/newname.fld complete"

File
$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/sv2fld.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/sv2fld.z

INDERS3 document set
Fusion
2 Converters
2.9 img2img

2.9 img2img

Synopsis
convert one image type to another image type

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with convert man pages
Submit	UIbutton	spawn a window to execute convert
Active Edit	UItoggle	if set, allow edit of convert parameters
Filename	UItext	input filename as passed to convert

Output Ports

none

Description

GUI interface for convert external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports
Input_Dir
A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory
A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters
File Name
UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file can be any image format (see convert man pages).

Output File
UIfileSB file browser. Selects the location and name of the disk file to output. The extension defines the type of image to convert to.

The output file can be any image extension (see convert man pages).

Help
Ulbutton. Pressing this button causes a new window to be spawned containing the man pages for convert.

Submit
Ulbutton. Pressing this button causes a new window to be spawned, in which the actual convert code is executed. Output from convert will appear in this window. The Submit button is only active when all necessary parameters for convert have been supplied.

Active Edit
Ultoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

filename
Ultext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

Interface to External Code
The external code convert is invoked using the shell script $HOME/INDERS_queue. The code convert reads from <filename.extension> and creates an AVS field file named: <filename.imgext>, which is then copied to the desired output filename.

Example Contents of $HOME/INDERS_queue
convert /rgbdata/abc.rgb /tiffdata/newname.tif
inform "/tiffdata/newname.tif complete"

File
$TOPDIR/inders3.project/v/converters.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/convert.z

INDERS3 document set
Fusion
2 Converters
2.10 luis2drs

2.10luis2drs

Synopsis
convert from UltraOptec's LUIS ultrasonic data file format to EPRI DRUS format. Optionally add a third coordinate extracted from a LUIS range data file.

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UlfileSB	name of the file to read
Output File	UlfileSB	name of the file to write
Help	Ulbutton	spawn a window with luis2drs man pages
Submit	Ulbutton	spawn a window to execute luis2drs
Active Edit	Ultoggle	if set, allow edit of luis2drs parameters
3D	Ultoggle	if set, call luis2drs2 following luis2drs
Filename	Ultext	input filename as passed to luis2drs

Output Ports

none

Description

GUI interface for luis2drs and luis2drs2 external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports
Input_Dir
A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory
A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name
UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is an UltraOptec's LUIS ultrasonic (.wav) format.

Output File
UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .drs, which will be appended if it is omitted.

The output file is an EPRI DRUS (.drs) format.

Help
UIbutton. Pressing this button causes a new window to be spawned containing the man pages for luis2drs.

Submit
UIbutton. Pressing this button causes a new window to be spawned, in which the actual luis2drs (and luis2drs2 when in 3D mode) code is executed. Output from one or both codes will appear in this window. The Submit button is only active when all necessary parameters for luis2drs have been supplied.

Active Edit
UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

3D
UItoggle. Activating this toggle causes luis2drs2 to be called following luis2drs. A LUIS range data file of the format <filename>.ran must be available in order to execute luis2drs2. May only be changed in Active Edit mode.

filename
UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

Interface to External Code
The external code luis2drs is invoked using the shell script $HOME/INDERS_queue. The code luis2drs reads from <filename.extension> and creates an AVS field file named: <filename>.drs, which is then copied to the desired output filename, unless the 3D option is selected.

If the 3D option is selected, the code luis2drs2 reads a LUIS range data file from <luisrangefilename>, which must have the extension .ran. luis2drs2 then copies from <luisrangefilename>.drs to <luisrangefilename>r.drs, inserting range data at the appropriate locations within the DRUS file. The range data represent the distance in LUIS specified units from the laser to the inspection point, so the first three DRUS coordinates in the output file describe a 3-D surface representing the inspected part.

Example Contents of $HOME/INDERS_queue

```
luis2drs /luisdata/960101.wav /luisdata/960101.drs
mv /luisdata/960101.drs /drsdata/newdata.drs
inform "/drsdata/newdata.drs complete"
```

Or if 3D option is set:

```
luis2drs /luisdata/960101.wav /luisdata/960101.drs
luis2drs2 /luisdata/960101.ran
mv /luisdata/960101r.drs /drsdata/newdata.drs
inform "/drsdata/newdata.drs complete"
```

This would produce a DRUS file describing the ultrasonic waveforms with respect to the part surface.

File
$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/luis2drs.f
-> $TOPDIR/inders3.project/src/inders3.codes/luis2drs2.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/luis2drs.z
-> $TOPDIR/inders/runtime/catman/a_man/cat1/luis2drs2.z

2.11 maus2fld

Synopsis
convert from MAUS ultrasonic waveform data format to AVS/Express field format

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output Directory	UIfileSB	name of the directory to write
Help	UIbutton	spawn a window with maus2rgb man pages
Submit	UIbutton	spawn a window to execute maus2rgb
Active Edit	UItoggle	if set, allow edit of maus2rgb parameters
trim	UItoggle	trim zero data in image buffer
downsampling factor	UIradioBox	downsampling factor when trim is set
Filename	UItext	input filename as passed to maus2rgb
switches	UItext	displays the selected switches

Output Ports

none

Description

GUI interface for maus2rgb external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports
Input_Dir
A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory
A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters
File Name
UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a MAUS ultrasonic data format.

Output Directory
UIfileSB directory browser. Selects the directory location for the disk files to output.

The output files are an AVS/Express field (.fld) format.

Help
UIbutton. Pressing this button causes a new window to be spawned containing the man pages for maus2rgb.

Submit
UIbutton. Pressing this button causes a new window to be spawned, in which the actual maus2rgb code is executed. Output from maus2rgb will appear in this window. The Submit button is only active when all necessary parameters for maus2rgb have been supplied.

Active Edit
UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

trim
UItoggle. Trim wasted space (zero data) in image buffer.

downsampling factor

UIradioBox. Chose downsampling factor when trim is set. Default is 1 when trim is not set.

filename

UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

switches

UItext. Displays the selected switches. This field should always contain "f". When this field only contains "f", there will be no trimming or downsampling. This should not be changed directly. To set trimming or downsampling please use the trim UItoggle and the downsampling factor UIradioBox.

Interface to External Code

The external code maus2rgb is invoked using the shell script $HOME/INDERS_queue. The code is invoked with an "f" switch, causing maus2rgb to create AVS/Express field files. The code maus2rgb reads from <filename.extension> and creates a series of AVS/Express .fld format files named <mausfilename>.XX.fld, where XX is the MAUS feature number (ie. 00, 01, 02, ...). These files are then moved to the directory specified in Output Directory.

Example Contents of $HOME/INDERS_queue

```
maus2rgb /mausdata/960202af f
mv /mausdata/969292af*.fld /flddata
inform "/flddata complete"
```

or with trimming and downsampling:

```
maus2rgb /mausdata/960202af ft5
mv /mausdata/960202af*.fld /flddata
inform "/flddata complete"
```

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/maus2rgb.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/maus2rgb.z

INDERS3 document set
Fusion
2 Converters
2.12 maus2rgb

2.12maus2rgb (NOT AVAILABLE ON PC)

Synopsis

convert from MAUS ultrasonic waveform data format to SGI RGB format

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output Directory	UIfileSB	name of the directory to write
Help	UIbutton	spawn a window with maus2rgb man pages
Submit	UIbutton	spawn a window to execute maus2rgb
Active Edit	UItoggle	if set, allow edit of maus2rgb parameters
trim	UItoggle	trim zero data in image buffer
downsampling factor	UIradioBox	downsampling factor when trim is set
Filename	UItext	input filename as passed to maus2rgb
switches	UItext	displays the selected switches

Output Ports

none

Description

GUI interface for maus2rgb external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a MAUS ultrasonic data format.

Output Directory

UIfileSB directory browser. Selects the directory location for the disk files to output.

The output files are in SGI RGB (.rgb) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for maus2rgb.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual maus2rgb code is executed. Output from maus2rgb will appear in this window. The Submit button is only active when all necessary parameters for maus2rgb have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

trim

UItoggle. Trim wasted space (zero data) in image buffer.

downsampling factor

UIradioBox. Chose downsampling factor when trim is set. Default is 1 when trim is not set.

filename

UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

switches

UItext. Displays the selected switches. This field should always contain "r". When this field only contains "r", there will be no trimming or downsampling. This should not be changed directly. To set trimming or downsampling please use the trim UItoggle and the downsampling factor UIradioBox.

Interface to External Code

The external code maus2rgb is invoked using the shell script $HOME/INDERS_queue. The code is invoked with an "r" switch, causing maus2rgb to create SGI RGB files. The code maus2rgb reads from <filename.extension> and creates a series of AVS/Express .fld format files named <mausfilename>.XX.rgb, where XX is the MAUS feature number (ie. 00, 01, 02, ...). These files are then moved to the directory specified in Output Directory.

Example Contents of $HOME/INDERS_queue

maus2rgb /mausdata/960202af r
mv /mausdata/969292af*.rgb /rgbdata
inform "/rgbdata complete"

or with trimming and downsampling:

maus2rgb /mausdata/960202af rt5
mv /mausdata/960202af*.rgb /rgbdata
inform "/rgbdata complete"

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/maus2rgb.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/maus2rgb.z

INDERS3 document set
Fusion
2 Converters
2.13 nastran2ucd

2.13nastran2ucd

Synopsis
convert from NASTRAN input file to AVS/Express UCD format.

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with nastran2ucd man pages
Submit	UIbutton	spawn a window to execute nastran2ucd
Active Edit	UItoggle	if set, allow edit of nastran2ucd parameters
b	UItoggle	if set, make FORTRAN BLOCKDATA
e	UItoggle	if set, echo NASTRAN file
Filename	UItext	input filename as passed to nastran2ucd
switches	UItext	displays the selected switches

Output Ports

none

Description

GUI interface for nastran2ucd external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports
Input_Dir
A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory
A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters
File Name
UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a NASTRAN input format.

Output File
UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .ucd, which will be appended if it is omitted.

The output file is an AVS/Express UCD (.ucd) format.

Help
UIbutton. Pressing this button causes a new window to be spawned containing the man pages for nastran2ucd.

Submit
UIbutton. Pressing this button causes a new window to be spawned, in which the actual nastran2ucd code is executed. Output nastran2ucd will appear in this window. The Submit button is only active when all necessary parameters for nastran2ucd have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

b (make FORTRAN BLOCKDATA)
UItoggle. Activating this toggle causes nastran2ucd to make FORTRAN BLOCKDATA source representing object geometry named <nastranfilename>.f in addition to <nastranfilename>.ucd. May only be changed in Active Edit mode.

e (echo NASTRAN file)
UItoggle. Activating this toggle causes nastran2ucd to echo the NASTRAN file during processing. May only be changed in Active Edit mode.

filename
UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

switches
UItext. Displays the selected switches. This should not be changed directly. Any changes should be made using the "b" or "e" UItoggles.

Interface to External Code
The external code nastran2ucd is invoked using the shell script $HOME/INDERS_queue. The code nastran2ucd reads from <filename.extension> and creates an AVS/Express UCD file named: <filename>.ucd, which is then copied to the desired output filename.

Only the following NASTRAN elements are recognized:

GRID, CBAR, CONROD, CSHEAR, PBAR, CORD1R, CORD2R, MAT1, PSHEAR

Example Contents of $HOME/INDERS_queue
nastran2ucd /nasdata/b1_90in_door be
mv /nasdata/b1_90in_door.ucd /ucddata/newdata.ucd
inform "/ucddata/newdata.ucd complete"

File
$TOPDIR/inders3.project/v/converters.v

See also
-> $TOPDIR/inders3.project/src/inders3.codes/nastran2ucd.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/nastran2ucd.z

INDERS3 document set
Fusion
2 Converters
2.14 neut2ydl

2.14 neut2ydl

Synopsis
read INDERS V1 CT neutral file format representing a volume CT set and extract a visualizable YOADL object

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with neut2ydl man pages
Submit	UIbutton	spawn a window to execute neut2ydl and neut2ydl2
Active Edit	UItoggle	if set, allow edit of neut2ydl parameters
CT Density	UIfield	CT density of YAODL surface model
r11	UIfield	1st coordinate of 1st line segment
r12	UIfield	2nd coordinate of 1st line segment
r13	UIfield	3rd coordinate of 1st line segment
r21	UIfield	1st coordinate of 2nd line segment
r22	UIfield	2nd coordinate of 2nd line segment
r23	UIfield	3rd coordinate of 2nd line segment
Number of Cylinders	UIfield	number of bounding cylinders

radiusN	UIfield	radius of bounding cylinder N
redN	UIfield	red component of bounding cylinder N
greenN	UIfield	green component of bounding cylinder N
blueN	UIfield	blue component of bounding cylinder N
rationalize	UItoggle	rationalize
tolerance	UIfield	tolerance value
Filename	UItext	input filename as passed to neut2ydl
CT density	UItext	CT density as passed to neut2ydl
line seg 1	UItext	line segment 1 as passed to neut2ydl
line seg 2	UItext	line segment 2 as passed to neut2ydl
cylinders	UItext	bounding cylinders as passed to neut2ydl
tolerance	UItext	tolerance as passed to neut2ydl2

Output Ports

none

Description

GUI interface for neut2ydl external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a CT neutral file.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .fld, which will be appended if it is omitted.

The output file is a YAODL (.ydl) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for neut2ydl.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual neut2ydl code and neut2ydl2 codes are executed. Output from both codes will appear in this window. The Submit button is only active when all necessary parameters for neut2ydl have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

CT Density

UIfield. CT density threshold of YAODL surface model

r11

UIfield. 1st coordinate of 1st line segment (see Interface to External Code).

r12

UIfield. 2nd coordinate of 1st line segment (see Interface to External Code).

r13

UIfield. 3rd coordinate of 1st line segment (see Interface to External Code).

r21

UIfield. 1st coordinate of 2nd line segment (see Interface to External Code).

r22

UIfield. 2nd coordinate of 2nd line segment (see Interface to External Code).

r23

UIfield. 3rd coordinate of 2nd line segment (see Interface to External Code).

Number of Cylinders

UIfield. Number of bounding cylinders. Makes radius, red, green, and blue UIfields visible for each cylinder requested.

radiusN

UIfield. Radius of bounding cylinder N (see Interface to External Code)

redN

UIfield. Red component of bounding cylinder N (see Interface to External Code)

greenN

UIfield. Green component of bounding cylinder N (see Interface to External Code)

blueN

UIfield. Blue component of bounding cylinder N (see Interface to External Code)

rationalize

UItoggle. Rationalize in neut2ydl2 as described in Interface to External Code)

tolerance

UIfield. If rationalize is not set, value is used as an option selector:

ltoll=1 simply reassembles the fragments. Ie. it has the same effect as

cat COORDS NORMAL COLORS CONCTY > temp.ydl

ltoll=2 adds a smoothing calculation (adjusts nodes to average of neighbors)
ltoll=3 only eliminates unconnected nodes but not redundant (ie. coplanar ones)

filename

UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

CT density

UItext. Displays the CT density as passed to neut2ydl. This should not be changed directly. Any changes should be made using the "CT Density" UIfield.

line seg 1

UItext. Displays the first line segment information as passed to neut2ydl. This should not be changed directly. Any changes should be made using the "r11", "r12", and "r13" UIfields.

line seg 2

UItext. Displays the second line segment information as passed to neut2ydl. This should not be changed directly. Any changes should be made using the "r21", "r22", and "r23" UIfields.

cylinders

UItext. Displays the bounding cylinder information as passed to neut2ydl. This should not be changed directly. Any changes should be made using the radius, red, green, and blue UIfields.

tolerance

UItext. Displays the rationalization and tolerance as passed to neut2ydl2. If rationalization is set, this number will be positive. If rationalization is not set, thei number will be -1*tolerance. This should not be changed directly. Any changes should be made using the rationalization UItoggle and tolerance UIfield.

Interface to External Code

The external codes neut2ydl and neut2ydl2 are invoked using the shell script $HOME/INDERS_queue. The code neut2ydl reads a CT neutral file and builds a YAODL surface model of the solution to the equation:

"CT density" = t

In other words, if the CT scanned part has a CT density of 2.5 g/cc and a threshold of t = 1.25 g/cc waselected, the object would represent the surface of the part.

If rij are present, the object is transformed to the midpoint of the line segment joining R1 and R2 and color coded by bounding cylinders whose axis is the line segment. The bounding cylinders are defined by their radius (ti1) and the RGB color (ti2,ti3,ti4). An initial bounding cylinder of radius zero and a final one of radius 1.0E30 are always implied.

If ti2 is less than zero (invalid RGB entry), the bounded object elements are excluded from the model rather than colored (ti2,ti3,ti4).

If R1 and R2 are coincident, the "ti1"s are interpreted as the distance from the plane "z = 0", rather than radii from R1-R2.

The surface areas of each of the color coded subsurfaces and the total surface area of the object will be printed.

neut2ydl is used typically for CT reverse engineering of test objects such as gears, etc. The color coding options allow subareas to be measured and displayed independently.

Normally, neut2ydl is followed by neut2ydl2 which allows more complete subarea partitioning options, and object rationalization. Because of this, the YAODL results are left in four files (COORDS, CONCTY, NORMAL, and COLORS).

neut2ydl2 reads the YAODL file fragments left by neut2ydl and rationalizes the YAODL object according to tol. If tol is negative, its absolute value is used as an option selector:

|tol|=1 simply reassembles the fragments. Ie. it has the same effect as

cat COORDS NORMAL COLORS CONCTY > temp.ydl

|tol|=2 adds a smoothing calculation (adjusts nodes to average of neighbors)
|tol|=3 only eliminates unconnected nodes but not redundant (ie. coplanar ones). This is a very quick operation compared to full rationalization. It is commonly used in conjunction when portions of the object were omitted (assigned a color of 'delete' in the prior neut2ydl step

If tol is zero or positive, neut2ydl2 rationalizes the object. Rationalizing steps are as follows:

ELIMINATING UNCONNECTED NODES
ELIMINATING REDUNDANT NODES {
CREATE NODE BASED STAR TABLE
SMOOTH (IF OPTION SELECTED)
ELIMINATE CENTER NODE OF STAR AND RETRIANGULATE IF AREA IS WITHIN tol OF BEING FLAT (ZERO CURVATURE)
ELIMINATING UNCONNECTED NODES }

The idea is to arrive at a simpler model representing the same geometry.

This code takes a lot of time on an object with many nodes (ie. many hours).

Example Contents of $HOME/INDERS_queue

```
neut2ydl /mctdata/GEAR.32 750
neut2ydl2 -1
mv temp.ydl /ydldata/newdata.ydl
inform "/ydldata/newdata.ydl complete"

neut2ydl GEAR.32 750 2,2,0 2,2,4 0.7,1,0,0 1.4,0,0.5,0.5
neut2ydl2 -3
mv temp.ydl /ydldata/newdata.ydl
inform "/ydldata/newdata.ydl complete"
```

File
$TOPDIR/inders3.project/v/converters.v

See also
-> $TOPDIR/inders3.project/src/inders3.codes/neut2ydl.f
-> $TOPDIR/inders3.project/src/inders3.codes/neut2ydl2.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/neut2ydl.z
-> $TOPDIR/inders/runtime/catman/a_man/cat1/neut2ydl2.z

INDERS3 document set
Fusion
2 Converters
2.15 neut2ucd

2.15 neut2ucd

Synopsis
read INDERS V1 CT neutral file format representing a volume CT set and extract a visualizable YOADL object. There is an option to add wall thickness to a YAODL file fragments model created using ydl2cncc, using neut2ydl3. The information is then converted into an AVS/Express UCD (*.ucd) format using ydl2ucd.

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with neut2ydl man pages
Submit	UIbutton	spawn a window to execute neut2ydl, neut2ydl2, ydl2cncc, neut2ydl3, and ydl2ucd
Active Edit	UItoggle	if set, allow edit of neut2ydl parameters
CT Density	UIfield	CT density of YAODL surface model
r11	UIfield	1st coordinate of 1st line segment
r12	UIfield	2nd coordinate of 1st line segment
r13	UIfield	3rd coordinate of 1st line segment
r21	UIfield	1st coordinate of 2nd line segment
r22	UIfield	2nd coordinate of 2nd line segment
r23	UIfield	3rd coordinate of 2nd line segment
Number of Cylinders	UIfield	number of bounding cylinders
radiusN	UIfield	radius of bounding cylinder N
redN	UIfield	red component of bounding cylinder N
greenN	UIfield	green component of bounding cylinder N
blueN	UIfield	blue component of bounding cylinder N
rationalize	UItoggle	rationalize
tolerance	UIfield	tolerance value
Add Wall Thickness	UItoggle	call neut2ydl3 to add wall thickness
otl	UIfield	surface normal tolerance value
thickmin	UIfield	lower limit of thicknesses to include
thickmax	UIfield	upper limit of thicknesses to include
Filename	UItext	input filename as passed to neut2ydl
CT density	UItext	CT density as passed to neut2ydl
line seg 1	UItext	line segment 1 as passed to neut2ydl
line seg 2	UItext	line segment 2 as passed to neut2ydl
cylinders	UItext	bounding cylinders as passed to neut2ydl
tolerance	UItext	tolerance as passed to neut2ydl2
thickness	UItext	thickness information as passed to neut2ydl3

Output Ports

none

Description

GUI interface for neut2ydl and neut2ydl3 external codes. The GUI allows entry of all necessary parameters and then spawns a separate window to run the codes upon the user's request. The user is informed when the codes have completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a CT neutral file.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .ucd, which will be appended if it is omitted.

The output file is an AVS/Express UCD (.ucd) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for neut2ydl.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual neut2ydl, neut2ydl2, ydl2cncc, neut2ydl3, and ydl2ucd codes are executed. Output from all codes will appear in this window. The Submit button is only active when all necessary parameters for neut2ydl and neut2ydl3 have been supplied.

Active Edit
UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

CT Density
UIfield. CT density threshold of YAODL surface model

r11 to r23
See neut2ydl.

Number of Cylinders
UIfield. Number of bounding cylinders. Makes radius, red, green, and blue UIfields visible for each cylinder requested.

radiusN, redN, greenN, blueN, rationalize, and tolerance
See neut2ydl.

filename
UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

CT density
UItext. Displays the CT density as passed to neut2ydl. This should not be changed directly. Any changes should be made using the "CT Density" UIfield.

line seg 1
UItext. Displays the first line segment information as passed to neut2ydl. This should not be changed directly. Any changes should be made using the "r11", "r12", and "r13" UIfields.

line seg 2
UItext. Displays the second line segment information as passed to neut2ydl. This should not be changed directly. Any changes should be made using the "r21", "r22", and "r23" UIfields.

cylinders
UItext. Displays the bounding cylinder information as passed to neut2ydl. This should not be changed directly. Any changes should be made using the radius, red, green, and blue UIfields.

tolerance
UItext. Displays the rationalization and tolerance as passed to neut2ydl2. If rationalization is set, this number will be positive. If rationalization is not set, thei number will be -1*tolerance. This should not be changed directly. Any changes should be made using the rationalization UItoggle and tolerance UIfield.

Interface to External Code

The external codes neut2ydl, neut2ydl2 ydl2cncc, neut2ydl3, and ydl2ucd are invoked using the shell script $HOME/INDERS_queue. The code neut2ydl reads a CT neutral file and builds a YAODL surface model of the solution to the equation:

"CT density" = t

See neut2ydl for a further description.

Normally, neut2ydl is followed by neut2ydl2 which allows more complete subarea partitioning options, and object rationalization. See neut2ydl for a further description.

ydl2cncc reads the YAODL file created by neut2ydl2 and writes the four files COORDS, NORMAL, COLORS, and CONCTY. These are referred to as the YAODL fragments. The following command will recreate the original YAODL file:

```
cat COORDS NORMAL COLORS CONCTY > filename.ydl
```

If the Add Wall Thickness option is select neut2ydl3 is then called. The code neut2ydl3 reads the YAODL file fragments and writes COLORS.tmp, a replacement file for COLORS which encodes the three element features described below.

Wall thickness in centimeters becomes the first element feature. Wall thicknesses outside the range thkmin:thkmax are set to zero. The second feature is the same thing in inches, and the third feature is the same as the first except that only thicknesses above thkmax are set to zero (mainly used for debugging or tuning thickness parameter selections). Since the algorithm for calculating thickness uses the surface normal to determine admissable facets to calculate distances between, a tolerance is specified for admissability (otl). Admissable facet pairs for thickness estimation satify the equation:

(n1.n2) < (-1.0+otl)

where n1 and n2 are the facet normal unit vectors.

Finally, ydl2ucd is called to create the UCD file. First COLORS.tmp is moved to COLORS (if in Add Wall Thickness mode). The code ydl2ucd reads the YAODL file fragments and writes an AVS/Express UCD file named newfile.ucd. Elements whose centers are not inside the boundary surface defined by the closed path defined by the pointlist x1,y1,...,xnc,ync are not put out, enabling portions of the model to be truncated.

Example Contents of $HOME/INDERS_queue
neut2ydl /mctdata/GEAR.32 750
neut2ydl2 -1
ydl2cncc temp.ydl
neut2ydl3 0.5,0.010,0.060
cp COLORS.tmp COLORS
ydl2ucd newdata.ucd 4,0.0,0.0,3.0,0.0,3.0,3.0,0.0,3.0
inform "/ucddata/newdata.ucd complete"

File
$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/neut2ydl.f
-> $TOPDIR/inders3.project/src/inders3.codes/neut2ydl2.f
-> $TOPDIR/inders3.project/src/inders3.codes/ydl2cncc.f
-> $TOPDIR/inders3.project/src/inders3.codes/neut2ydl3.f
-> $TOPDIR/inders3.project/src/inders3.codes/ydl2ucd.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/neut2ydl.z
-> $TOPDIR/inders/runtime/catman/a_man/cat1/neut2ydl2.z
-> $TOPDIR/inders/runtime/catman/a_man/cat1/ydl2cncc.z
-> $TOPDIR/inders/runtime/catman/a_man/cat1/neut2ydl3.z
-> $TOPDIR/inders/runtime/catman/a_man/cat1/ydl2ucd.z

2.16ps2drs

Synopsis
convert from ShowCase digitized graph data (PostScript) into EPRI DRUS format.

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with ps2drs man pages
Submit	UIbutton	spawn a window to execute ps2drs
Active Edit	UItoggle	if set, allow edit of ps2drs parameters
p	UItoggle	if set, plot using plotint
x	UItoggle	if set, send plot to XZ window
Filename	UItext	input filename as passed to ps2drs
switches	UItext	displays the selected switches

Output Ports

none

Description

GUI interface for the ps2drs external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports
Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a ShowCase digitized graph data - PostScript (.ps) format.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .drs, which will be appended if it is omitted.

The output file is an EPRI DRUS (.drs) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for ps2drs.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual ps2drs code is executed. Output from ps2drs code will appear in this window. The Submit button is only active when all necessary parameters for ps2drs have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

p

UItoggle. If set, plot the results (using plotint).

x

UItoggle if set, send the plotted results to the XZ-window display, not a PostScript printer.

filename

UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

switches

UItext. Displays the selected switches. This should not be changed directly. Any changes should be made using the "p" or "x" UItoggles.

Interface to External Code

The external code ps2drs is invoked using the shell script $HOME/INDERS_queue. The code ps2drs reads a PostScript file written by ShowCase and creates a DRUS file named: <filename>.DRS, which is then copied to the desired output filename.

The code ps2drs interprets line drawing (PostScript) commands into rasterized DRUS signals. Typically this is used for digitizing graph data such as radar data (response .vs. frequency) which has been scanned in on a desktop scanner from non-digital (paper) records.

Example Contents of $HOME/INDERS_queue

ps2drs /psdata/0005R.index
mv /psdata/0005R.DRS /drsdata/newdata.drs
inform "/drsdata/newdata.drs complete"

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/ps2drs.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/ps2drs.z

INDERS3 document set
Fusion
2 Converters
2.17 sdt2ucd

2.17 sdt2ucd

Synopsis
convert WinSpect format to AVS/Express UCD format

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with sdt2ucd man pages
Submit	UIbutton	spawn a window to execute sdt2ucd
Active Edit	UItoggle	if set, allow edit of sdt2ucd parameters
Filename	UItext	input filename as passed to sdt2ucd

Output Ports

none

Description

GUI interface for sdt2ucd external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports
Input_Dir
A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory
A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters
File Name
UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a WinSpect format (.sdt) format.

Output File
UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .ucd, which will be appended if it is omitted.

The output file is an AVS/Express UCD (.ucd) format.

Help
UIbutton. Pressing this button causes a new window to be spawned containing the man pages for sdt2ucd.

Submit
UIbutton. Pressing this button causes a new window to be spawned, in which the actual sdt2ucd code is executed. Output from sdt2ucd will appear in this window. The Submit button is only active when all necessary parameters for sdt2ucd have been supplied.

Active Edit
UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

filename
UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

Interface to External Code
The external code sdt2ucd is invoked using the shell script $HOME/INDERS_queue. The code sdt2ucd reads from <filename.extension> and creates an AVS/Express YCD file named: <filename>.ucd, which is then copied to the desired output filename.

Example Contents of $HOME/INDERS_queue
sdt2drs /sdtdata/trisomy.sdt
mv /sdtdata/trisomy.ucd /ucddata/newname.ucd

inform "/ucddata/newname.ucd complete"

File
$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/sdt2ucd.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/sdt2ucd.z

2.18 sonix2drs

Synopsis
convert from Sonix scanner data format into EPRI DRUS format.

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with sonix2drs man pages
Submit	UIbutton	spawn a window to execute sonix2drs
Active Edit	UItoggle	if set, allow edit of sonix2drs parameters
Image Number	UIfield	index number of image to extract
Filename	UItext	input filename as passed to sonix2drs
switches	UItext	displays the selected switches

Output Ports

none

Description

GUI interface for the sonix2drs external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports
Input_Dir
A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory
A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters
File Name
UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a Sonix scanner data format.

Output File
UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .drs, which will be appended if it is omitted.

The output file is an EPRI DRUS (.drs) format.

Help
UIbutton. Pressing this button causes a new window to be spawned containing the man pages for sonix2drs.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual sonix2drs code is executed. Output from sonix2drs code will appear in this window. The Submit button is only active when all necessary parameters for sonix2drs have been supplied.

Active Edit
UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

Image Number
UIfield. The index number of the image to extract from the input file, where the first image has an index number of 1. A value of 0 is equivalent to ALL. A value greater than the number of scans will cause the program to stop. The default value is 0 (all images).

filename
UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

switches
UItext. Displays the selected switches. This should not be changed directly. Any changes should be made using the "Image Number" UIfield.

Interface to External Code
The external code sonix2drs is invoked using the shell script $HOME/INDERS_queue. The code sonix2drs reads Sonix Proprietary Data file and creates a DRUS file with the chosen output name.

The code sonix2drs recognizes both Sonix Proprietary Data Format and TIFF File Format. The output file is ASCII hexidecimal (two characters per pixel) biased by 80 (hex) in lieu of signed values.

Example Contents of $HOME/INDERS_queue
sonix2drs /sonixdata/ w100_3-1.rf /drsdata/newfile.drs
inform "/drsdata/newdata.drs complete"

File
$TOPDIR/inders3.project/v/converters.v

See also
-> $TOPDIR/inders3.project/src/inders3.codes/sonix2drs.c
-> $TOPDIR/inders3.project/src/inderslib/asdrusBoeing.c
-> $TOPDIR/inders3.project/src/inderslib/asneeded.c
-> $TOPDIR/inders3.project/src/inderslib/asutile.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/sonix2drs.z

INDERS3 document set
Fusion
2 Converters
2.19 step2ucd

2.19step2ucd

Synopsis
convert from STEP geometry model into AVS/Express UCD (.ucd) format.

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with step2ucd man pages
Submit	UIbutton	spawn a window to execute step2ucd
Active Edit	UItoggle	if set, allow edit of step2ucd parameters
d	UItoggle	if set, debug information is printed
Filename	UItext	input filename as passed to step2ucd
switches	UItext	displays the selected switches

Output Ports

none

Description

GUI interface for the step2ucd external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a STEP geometry file.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .ucd, which will be appended if it is omitted.

The output file is an AVS/Express UCD (.ucd) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for step2ucd.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual step2ucd code is executed. Output from step2ucd code will appear in this window. The Submit button is only active when all necessary parameters for step2ucd have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

d

UItoggle. If set, tracing (debug) information is printed during processing.

filename

UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

switches

UItext. Displays the selected switches. This should not be changed directly. Any changes should be made using the "d" UItoggle.

Interface to External Code

The external code step2ucd is invoked using the shell script $HOME/INDERS_queue. The code step2ucd reads a STEP geometry model (ISO-10303-21) and converts to an AVS/Express UCD format file named: <filename>.ucd, which is then copied to the desired output filename. The presence of the switch "u" indicates to step2ucd that a UCD file should be created.

Example Contents of $HOME/INDERS_queue

step2ucd /stepdata/14FOOTXFER.step u
mv /stepdata/14FOOTXFER.ucd /ucddata/newdata.ucd
inform "/ucddata/newdata.ucd complete"

File

$TOPDIR/inders3.project/v/converters.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/step2ucd.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/step2ucd.z

INDERS3 document set
Fusion
2 Converters
2.20 step2ydl

2.20step2ydl

Synopsis

convert from STEP geometry model into YAODL (.ydl) format.

Input Ports

Input_Dir	string	String containing input search directory path
Output_Directory	string	String containing output search directory path

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	spawn a window with step2ucd man pages
Submit	UIbutton	spawn a window to execute step2ucd
Active Edit	UItoggle	if set, allow edit of step2ucd parameters
d	UItoggle	if set, debug information is printed
Filename	UItext	input filename as passed to step2ucd
switches	UItext	displays the selected switches

Output Ports

none

Description

GUI interface for the step2ucd external code. The GUI allows entry of all necessary parameters and then spawns a separate window to run the code upon the user's request. The user is informed when the code has completed.

Input Ports

Input_Dir

A port to connect to a default directory path in which to find the input file. If unconnected, the browser will begin in the start-up directory.

Output_Directory

A port to connect to a default directory path in which to locate the output file. If unconnected, the browser will begin in the start-up directory.

Parameters

File Name

UIfileSB file browser. Selects the disk file to input and convert. The default search pattern is $Input_Dir/*.*.

The input file is a STEP geometry file.

Output File

UIfileSB file browser. Selects the location and name of the disk file to output. The extension is required to be .ydl, which will be appended if it is omitted.

The output file is YAODL (.ydl) format.

Help

UIbutton. Pressing this button causes a new window to be spawned containing the man pages for step2ucd.

Submit

UIbutton. Pressing this button causes a new window to be spawned, in which the actual step2ucd code is executed. Output from step2ucd code will appear in this window. The Submit button is only active when all necessary parameters for step2ucd have been supplied.

Active Edit

UItoggle. Activating this toggle allows editing of parameters. When off, all parameters are inactive.

d

UItoggle. If set, tracing (debug) information is printed during processing.

filename

UItext. Displays the selected input filename. This is for confirmation only. Any changes in input filename should be made using the File Name browser.

switches

UItext. Displays the selected switches. This should not be changed directly. Any changes should be made using the "d" UItoggle.

Interface to External Code
The external code step2ucd is invoked using the shell script $HOME/INDERS_queue. The code step2ucd reads a STEP geometry model (ISO-10303-21) and converts to a YAODL format file named: <filename>.ydl, which is then copied to the desired output filename. The presence of the switch "y" indicates to step2ucd that a YAODL file should be created.

Example Contents of $HOME/INDERS_queue
step2ucd /stepdata/14FOOTXFER.step y
mv /stepdata/14FOOTXFER.ydl /ydldata/newdata.ydl
inform "/ydldata/newdata.ydl complete"

File
$TOPDIR/inders3.project/v/converters.v

See also
-> $TOPDIR/inders3.project/src/inders3.codes/step2ucd.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/step2ucd.z

Geometries are modules that have been developed from part drawings. These allow visualization and manipulation of the part in AVS/Express. These may be used in conjunction with the Readers to map data onto a part surface.

3 Geometries

3.1 JSF_Spar

Synopsis
JSF specific INDERS3 geometry object to visualize a JSF spar.

Input Ports

None

Parameters

Wireframe	Ultoggle	If set, display wireframe only
Length	Ulfield	spar length
Width	Ulfield	spar width
Inner_Bond_Width	Ulfield	bond width
Spar_Depth	Ulfield	spar depth
Ply_Thickness	Ulfield	ply thickness
Num_Skin_Plies	Ulfield	number of skin plies
Num_Spar_Plies	Ulfield	number of spar plies
Pad_Width	Ulfield	pad width
Num_Pad_Plies	Ulfield	number of pad plies

Output Ports

obj	DefaultObject	provides a visualizable object to the Uviewer

Description

JSF_Spar has a Graphical User Interface (GUI) that allows the user to modify the geometry of the JSF spar to visualize. The user can also specify that the spar should be displayed as a wireframe by toggling "Wireframe". This incorporates the geometry generator code from JSF_52inSpar.

Parameters

Wireframe
Ultoggle. If set, display wireframe only

Length
Ulfield. Length of spar to visualize. Default is 52 inches.

Width
Ulfield. Width of spar to visualize. Default is 9 inches.

Inner_Bond_Width
Ulfield. Width of inner bond to visualize (where cross piece of spar is bonded). Default is 4.25 inches.

Spar_Depth
Ulfield. Depth of spar to visualize. Default is 4.5 inches.

Ply_Thickness
Ulfield. Thickness of each ply in spar. Default is .0055 inches.

Num_Skin_Plies
Ulfield. Number of skin plies. This determines skin depth by multiplying Num_Skin_Plies by Ply_Thickness. Default for Num_Skin_Plies is 42.

Num_Spar_Plies
Ulfield. Number of spar plies. This determines spar crossbar width by multiplying Num_Spar_Plies by Ply_Thickness. Default for Num_Skin_Plies is 10.

Pad_Width
Ulfield. Pad width. Default is 4.25 inches.

Num_Pad_Plies
Ulfield. Number of pad plies. This determines pad thicknes by multiplying Num_Pad_Plies by Ply_Thickness. Default for Num_Pad_Plies is 0 (no pad).

Output Ports
obj
Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer.

File
$TOPDIR/inders3.project/v/JSF.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_JSF_Spar.z
INDERS documentation set
Fusion
3 Geometries
3.2 AWACS Radome

3.2 AWACS_Radome

Synopsis
AWACS specific INDERS3 geometry object to visualize an AWACS Radome.

Input Ports

None

Parameters

None

Output Ports

obj	DefaultObject	provides a visualizable object to the Uviewer

Description

AWACS_Radome has no Graphical User Interface (GUI). This object is generated using the geometry code from AWACS_Radome.

The geometry code AWACS_Radome creates three files named AWACS_Radome.v, AWACS_RadomeChalkLines.v, and AWACS_RadomeChalkLabels.v. These describe the subcomponents of the AWACS Radome Geometry object (the surface shape, the "longitude" and "latitude" lines, and the "longitude" and "latitude" labels, respectively).

Output Ports
obj
Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer.

File
$TOPDIR/inders3.project/v/AWACS.v

See also
-> $TOPDIR/inders3.project/src/inders3.codes/AWACS_Radome.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_AWACS_Radome.z
-> $TOPDIR/inders/runtime/catman/a_man/cat1/AWACS_Radome.z
INDERS documentation set
Fusion
3 Geometries
3.3 B1 90inDoor

3.3 B1 90inDoor

Synopsis
B1 specific INDERS3 geometry object to visualize a B1 90" door.

Input Ports

None

Parameters

None

Output Ports

obj	DefaultObject	provides a visualizable object to the Uviewer

Description

B1_90inDoor has no Graphical User Interface (GUI). This object is generated using the geometry code from B1_90inDoor.

Output Ports
obj
Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer.

File
$TOPDIR/inders3.project/v/B1.v

See also
-> $TOPDIR/inders3.project/src/inders3.codes/B1_90inDoor.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_B1_90inDoor.z
-> $TOPDIR/inders/runtime/catman/a_man/cat1/B1_90inDoor.z
INDERS documentation set
Fusion
3 Geometries
3.4 B1 180inDoor

3.4 B1 180inDoor

Synopsis
B1 specific INDERS3 geometry object to visualize a B1 180" door.

Input Ports

None

Parameters

None

Output Ports

obj	DefaultObject	provides a visualizable object to the Uviewer

Description

B1_180inDoor has no Graphical User Interface (GUI). This object is generated using the geometry code from B1_180inDoor.

Output Ports
obj
Provides the visualizable object to the Uviewer. To manipulate in 3D, connect to a 3D Uviewer.

File
$TOPDIR/inders3.project/v/B1.v

See also

-> $TOPDIR/inders3.project/src/inders3.codes/B1_180inDoor.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_B1_180inDoor.z
-> $TOPDIR/inders/runtime/catman/a_man/cat1/B1_180inDoor.z

INDERS documentation set
Fusion
4 Macros

Macros are modules that are used within various other modules. They are the backbone to construct complex data visualization modules.

4 Macros

INDERS documentation set
Fusion
4 Macros
4.1 INDERS general

4.1 INDERS general

INDERS documentation set
Fusion
4 Macros
4.1 INDERS general4.1.1 DisableNullStringJournalError

4.1.1 DisableNullStringJournalError

Synopsis
INDERS macro prevent AVS/Express from displaying the "Null String Journal" error.

Input Ports

None

Parameters

None

Output Ports

None

Description

AVS/Express prints a "Null String Journal" error at times. This is a meaningless error, but causes the Error window to come up. Including this module in your application prevents this. It is automatically included in the Startup Application.

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_DisableNullStringJournalError.z
INDERS documentation set
Fusion
4 Macros
4.1 INDERS general4.1.2 fizTitle

4.1.2 fizTitle

Synopsis
INDERS macro to pass a string through.

Input Ports

string1	&string	string to recast

Parameters

None

Output Ports

title	string	recast of string1

Description

At times, AVS/Express will not pass strings on. In that case, try placing a fizTitle component between the string and the input to which you are passing it. It will be copied to title using a parse_v command.

Input Ports
string1
Reference to a string to be recast.

Output Ports
title
Recasting of string1.

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_fizTitle.z
INDERS documentation set
Fusion
4 Macros
4.1 INDERS general4.1.3 get_coords

4.1.3 get_coords

Synopsis
INDERS macro to get the coordinates from a mesh.

Input Ports

in	&mesh	mesh to extract coordinates from

Parameters

None

Output Ports

coord	olink	coordinate passed from DVxform_coord

Description

Passes the mesh into DVxform_coord which extracts the first 3 coordinates.

Input Ports
in
Mesh to extract coordinates from

Output Ports
coord
Passed from DVxform_coord.

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_get_coords.z
INDERS documentation set
Fusion
4 Macros
4.1 INDERS general4.1.4 get_instance

4.1.4 get_instance

Synopsis
INDERS macro to get the instance number of an object

Input Ports

None

Parameters

None

Output Ports

instance_number	int	instance number of object

Description

Retrieves the instance number of any object in which it is instanced. Reads the value after the "" sign in the parent object and returns it in instance_number. Returns zero if there is no "" sign.

Output Ports
instance_number
Instance number of object in which get_instance is instanced.

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_get_instance.z
INDERS documentation set
Fusion
4 Macros
4.1 INDERS general4.1.5 Indir_Outfile

4.1.5 Indir_Outfile

Synopsis

INDERS macro to provide a GUI to provide an input directory and an output filename, plus the "Allow Edit" UItoggle, the "Submit" UIbutton, and the "Help" UIbutton.

Input Ports

parent	UIconnection port to connect to parent UImod_panel	
label	string	label the top of the GUI
Input_Dir	string	directory to begin input search
Output_Directory	string	directory to begin output search
SUBMIT.do	int	resets the SUBMIT UIbutton
HELP.do	int	resets the HELP UIbutton
SUBMIT.active	int	sets SUBMIT UIbutton active
HELP.active	int	sets HELP UIbutton active

Parameters

Directory	UIfileSB	name of the directory to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	output passed outside module
Submit	UIbutton	output passed outside module
Active Edit	UItoggle	output passed outside module

Output Ports

HELP.do	int	HELP UIbutton output
SUBMIT.active	int	SUBMIT UIbutton output
Allow_Edit	int	if set, Allow Edit is toggled on

Description

Provides a UIfileSB directory browser for input and a UIfileSB file browser for output. Files and directories are checked for validity and seperated into path names and filenames, which are available using the Network Editor.

Input Ports

parent

Port to connect to parent UImod_panel, UIpanel or UIframe.

label

label the top of the GUI

Input_Dir

directory to begin input search

Output_Directory

directory to begin output search

SUBMIT.do

Resets the SUBMIT UIbutton. Allows the application engineer to determine when SUBMIT should be reset.

HELP.do

Resets the HELP UIbutton. Allows the application engineer to determine when HELP should be reset.

SUBMIT.active

Sets SUBMIT UIbutton active. Allows the application engineer to determine when SUBMIT should be active.

HELP.active

sets HELP UIbutton active. Allows the application engineer to determine when HELP should be active.

Parameters

Directory

UIfileSB . Name of the directory to read.

Output File

UIfileSB . Name of the file to write.

HELP

UIbutton. Output passed out of module. Reset and active controlled by inputs to module.

SUBMIT

UIbutton. Output passed out of module. Reset and active controlled by inputs to module.

Active Edit
UItoggle. Output passed out of module. Active controlled by valid presence of a valid input directory.

Output Ports
HELP.do
HELP UIbutton output

SUBMIT.active
SUBMIT UIbutton output

Allow_Edit
if set, Allow Edit is toggled on

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Indir_Outfile.z

INDERS documentation set
Fusion
4 Macros
4.1 INDERS general4.1.6 Infile_Outdir

4.1.6 Infile_Outdir

Synopsis
INDERS macro to provide a GUI to provide an input file and an output directory, plus the "Allow Edit" UItoggle, the "Submit" UIbutton, and the "Help" UIbutton.

Input Ports

parent	UIconnection port to connect to parent UImod_panel	
label	string	label the top of the GUI
Input_Dir	string	directory to begin input search
Output_Directory	string	directory to begin output search
SUBMIT.do	int	resets the SUBMIT UIbutton
HELP.do	int	resets the HELP UIbutton
SUBMIT.active	int	sets SUBMIT UIbutton active
HELP.active	int	sets HELP UIbutton active

Parameters

File Name	UIfileSB	name of the file to read
Output Directory	UIfileSB	name of the directory to write
Help	UIbutton	output passed outside module
Submit	UIbutton	output passed outside module
Active Edit	UItoggle	output passed outside module

Output Ports

HELP.do	int	HELP UIbutton output
SUBMIT.active	int	SUBMIT UIbutton output
Allow_Edit	int	if set, Allow Edit is toggled on

Description

Provides a UIfileSB file browser for input and a UIfileSB directory browser for output. Files and directories are checked for validity and seperated into path names and filenames, which are available using the Network Editor.

Input Ports
parent
Port to connect to parent UImod_panel, UIpanel or UIframe.

label
label the top of the GUI

Input_Dir
directory to begin input search

Output_Directory
directory to begin output search

SUBMIT.do
Resets the SUBMIT UIbutton. Allows the application engineer to determine when SUBMIT should be reset.

HELP.do
Resets the HELP UIbutton. Allows the application engineer to determine when HELP should be reset.

SUBMIT.active
Sets SUBMIT UIbutton active. Allows the application engineer to determine when SUBMIT should be active.

HELP.active
sets HELP UIbutton active. Allows the application engineer to determine when HELP should be active.

Parameters
File Name
UIfileSB . Name of the file to read.

Output Directory
UIfileSB . Name of the directory to write.

HELP
UIbutton. Output passed out of module. Reset and active controlled by inputs to module.

SUBMIT
UIbutton. Output passed out of module. Reset and active controlled by inputs to module.

Active Edit
UItoggle. Output passed out of module. Active controlled by valid presence of a valid input file.

Output Ports
HELP.do
HELP UIbutton output

SUBMIT.active
SUBMIT UIbutton output

Allow_Edit
if set, Allow Edit is toggled on

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Infile_Outdir.z

INDERS documentation set
Fusion
4 Macros
4.1 INDERS general4.1.7 Infile_Outfile

4.1.7 Infile_Outfile

Synopsis
INDERS macro to provide a GUI to provide an input filename and an output filename, plus the "Allow Edit" UItoggle, the "Submit" UIbutton, and the "Help" UIbutton.

Input Ports

parent	UIconnection	port to connect to parent UImod_panel
label	string	label the top of the GUI
Input_Dir	string	directory to begin input search
Output_Directory	string	directory to begin output search
SUBMIT.do	int	resets the SUBMIT UIbutton
HELP.do	int	resets the HELP UIbutton
SUBMIT.active	int	sets SUBMIT UIbutton active
HELP.active	int	sets HELP UIbutton active

Parameters

File Name	UIfileSB	name of the file to read
Output File	UIfileSB	name of the file to write
Help	UIbutton	output passed outside module
Submit	UIbutton	output passed outside module
Active Edit	UItoggle	output passed outside module

Output Ports

HELP.do	int	HELP UIbutton output
SUBMIT.active	int	SUBMIT UIbutton output
Allow_Edit	int	if set, Allow Edit is toggled on

Description

Provides a UIfileSB file browser for input and a UIfileSB file browser for output. Files and directories are checked for validity and seperated into path names and filenames, which are available using the Network Editor.

Input Ports

parent

Port to connect to parent UImod_panel, UIpanel or UIframe.

label

label the top of the GUI

Input_Dir

directory to begin input search

Output_Directory

directory to begin output search

SUBMIT.do

Resets the SUBMIT UIbutton. Allows the application engineer to determine when SUBMIT should be reset.

HELP.do

Resets the HELP UIbutton. Allows the application engineer to determine when HELP should be reset.

SUBMIT.active

Sets SUBMIT UIbutton active. Allows the application engineer to determine when SUBMIT should be active.

HELP.active

sets HELP UIbutton active. Allows the application engineer to determine when HELP should be active.

Parameters

File Name

UIfileSB . Name of the file to read.

Output File

UIfileSB . Name of the file to write.

HELP

UIbutton. Output passed out of module. Reset and active controlled by inputs to module.

SUBMIT

UIbutton. Output passed out of module. Reset and active controlled by inputs to module.

Active Edit

UItoggle. Output passed out of module. Active controlled by valid presence of a valid input directory.

Output Ports

HELP.do

HELP UIbutton output

SUBMIT.active

SUBMIT UIbutton output

Allow_Edit

if set, Allow Edit is toggled on

File

$TOPDIR/inders3.project/v/Tools.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Indir_Outfile.z

INDERS documentation set
Fusion
4 Macros
4.1 INDERS general4.1.8 writeCSV

4.1.8 writeCSV

Synopsis

INDERS macro to provide a GUI to provide an inteface to the writecsv function.

Input Ports

in	field	input field or upper field
lower	field	lower field (if supplied)

Parameters

Image_Filename	UIfileSB	name of the file to read
Write File	UIbutton	trigger file write
Overwrite	UItoggle	if set, overwrite existing file
JSF_toggle	UItoggle	if set, use JSF_Mesh
Data Tolerance	UIfield	data tolerance

Output Ports

None

Description

Provides a UIfileSB file browser for input. Also allows input of the other parameters neede to run writecsv.

Input Ports

in

Input field, or upper field (if lower field is supplied).

lower

Lower field (if supplied).

Parameters

Image Filename

UIfileSB . Name of the file to read.

Write File

UIbutton. Triggers file writing.

Overwrite

UItoggle. Allow overwrite of an existing file if set.

JSF_Toggle

UItoggle. If set, use JSF_Mesh, otherwise use AWACS_CSC

Data Tolerance

UIfield. Tolerance to pass to writecsv.

File

$TOPDIR/inders3.project/v/Tools.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_writeCSV.z

INDERS documentation set
Fusion
4 Macros
4.2 Data_Macros

4.2 Data_Macros

4.2.1 File_Info
4.2.2 Label_Position
4.2.3 DataUI
4.2.4 Switchable_Datamap
INDERS documentation set
Fusion
4 Macros
4.2 Data_Macros
4.2.1 File_Info

4.2.1 File_Info

Synopsis

INDERS3 structure to pass information on two files and the parameters being visualized.

```
File_Info {
    filename1       string      full filename of the first file
    filename2       string      full filename of the second file
    justfile1       string      filename without path of first file
    justfile2       string      filename without path of second file
    feature_num     int         feature number being visualized
    feature_name    string      feature name being visualized
    units           string      units of feature being visualized
    xform           float[3]    where to place the data
    min             float       minimum of data pseudocolor
    max             float       maximum of data pseudocolor
    numIntervals    int         number of intervals in pseudocolor
    numTicks        int         number of ticks to place in legend
    selectedView    int         selected view
}
```

Description

Group values are filled by various GUIs and drus_preface and bfs_header. Values are then available to Plot Macros or to the user.

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_File_Info.z

INDERS documentation set
Fusion
4 Macros
4.2 Data_Macros
4.2.2 Label_Position

4.2.2 Label_Position

Synopsis

INDERS3 structure to pass information on labelling data.

```
Label_Position {
    x_min           float       minimum x position of label
    x_max           float       maximum x position of label
    y_min           float       minimum y position of label
    y_max           float       maximum y position of label
    z_val           float       z position of label
    title_position  float[3]    title postion (x, y, z)
    stoke           int         if set, use strokable text
    expansion       float       expansion of text
    height          float       height of text
    fixed           boolean     if set, fix text on page
    max_step_label  int         maximum number of steps in label
    max_cont_label  int         maximum labels on continuous pseudocolor
    Label_col       float[]     color to use for label
```

Title_col	float[]	color to use for title

}

Description

Group values are filled by the application engineer to determine visualization of legend, labels, and ticks.

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Label_Position.z
INDERS documentation set
Fusion
4 Macros
4.2 Data_Macros
4.2.3 DataUI

4.2.3 DataUI

Synopsis
INDERS macro to provide a GUI to provide an inteface to the Switchable_Datamap module.

Input Ports

UIpanel.y	float	Where to place the GUI
UIpanel.visible	int	if set, make GUI visible
UIpanel.parent	UIconnection	parent of GUI
Label_Position	&group	see Label_Position
DefaultMinMax.input\	field	field to pseudocolor

Parameters

Step_Colors	UItoggle	control continuous or stepped pseudocolor
Num_Steps	UIfield	number of steps in Step_Colors mode
Data_Limits	UItoggle	use data limits or Min and Max
Min	UIfield	min pseudocolor when Data Limits not set
Max	UIfield	max pseudocolor when Data Limits not set

Output Ports

None

Description

Provides a GUI to provide an inteface to the Switchable_Datamap module. Allows the user to chose whether pseudocolor should be stepped or continuous. If stepped is chosen, the user can chose the number of steps. The user can also chose whether the pseudocolor will be based on the data limits or on a user supplied minimum and maximum value.

Input Ports
UIpanel.y
Controls where the GUI is displayed within the parent.

UIpanel.visible
Controls visibility of GUI.

UIpanel.parent
Parent on which to display GUI.

Label_Position
Used to calculate the number of ticks to display.

DefaultMinMax.input
Input field to be colored. DefaultMinMax provides the minimum and maximum values of the data.

Parameters
Step_Colors
UItoggle. When set, pseudocolor is changed from 256 steps (continuous) to the value of Num_Steps. Default is not set (continuous colors).

Num_Steps
UIfield. Number of steps to use for pseudocolor. Only active when Step_Colors is set. Default is 8.

Data_Limits
UItoggle. Use data limits when set. When not set, use user supplied Min and Max. Default is set (use data limits).

Min
UIfield. If Data_Limits is not set, this is the minimum value used to pseudocolor.

Max
UIfield. If Data_Limits is not set, this is the maximum value used to pseudocolor.

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_DataUI.z

INDERS documentation set
Fusion
4 Macros
4.2 Data_Macros
4.2.4 Switchable_Datamap

4.2.4 Switchable_Datamap

Synopsis
Standard Datamap modified to use DataUI inputs to change pseudocolor limits and steps.

See Datamap for information on Ports and Parameters

Description

Modifies dataMin and dataMax as well as the DataRange size, based on the inputs from DataUI.

File
$TOPDIR/inders3.project/v/Tools.v

See also
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Switchable_Datamap.z

INDERS documentation set
Fusion
4 Macros
4.3 Plot_Macros

4.3 Plot_Macros

INDERS documentation set
Fusion
4 Macros
4.3 Plot_Macros
4.3.1 Data_Info

4.3.1 Data_Info

Synopsis

INDERS3 structure to pass information on which plots to display.

```
Data_Info {
    Mean_Rem_Plotboolean       if set, plot mean removed data
    Mean_Rem_Data          float[]      mean removed data array
    Log_Plot               boolean      if set, plot log decode data
    Log_Data      float[]       log decoded data array
    Env_Plot               boolean      if set, plot envelope
    Mean_Env_Datafloat[]        envelope of mean removed data
    Log_Env_Data           float[]      envelope of log decoded data
    Hide                   boolean      hide the plot
    rescale                int          minimum of data pseudocolor
}
```

Description

Group values are filled by various GUIs and mean_remove, log_decode and analytic_envalope. Values are then available to Plot Macros or to the user.

File
$TOPDIR/inders3.project/v/Tools.v

See also

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Data_Info.z

INDERS documentation set
Fusion
4 Macros
4.4 Extract_Macros

4.4 Extract_Macros

INDERS documentation set
Fusion
4 Macros
4.6 Dictionaries

4.6 Dictionaries

INDERS documentation set
Fusion
5 Miscellaneous

5 Miscellaneous

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions

5.1 Functions

5.1.1 String_functions

5.5.1.1 Strcmp

Synopsis

Function written in C to compare two strings. Equivalent to the C function strcmp.

Input Ports

String1	string	first string to compare
String2	string	second string to compare

Parameters

None

Output Ports

ret	int	comparison return

Description

Strcmp compares its arguments and returns an integer less than, equal to, or greater than 0, based upon whether String1 is lexicographically less than, equal to, or greater than String2.

Input Ports
String1
First of two strings to compare

String2
Second of two strings to compare.

Output Ports
ret
See Description above.

See also
-> $TOPDIR/inders3.project/src/Strcmp.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Strcmp.z

5.1.1.2 Strlen

Synopsis

Function written in C to find the length of a string. Equivalent to the C function strlen.

Input Ports

String	string	string to find length of

Parameters

None

Output Ports

Num_Char	int	number of characters, or zero if NULL

Description

Strlen returns the number of characters in String, not including the terminating null character. Returns zero if String is null.

Input Ports
String
String to find the length of.

Output Ports
Num_Char
Number of characters in String. Zero if String is null.

See also
-> $TOPDIR/inders3.project/src/Strlen.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Strlen.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.1 String functions
5.1.1.3 Strstr

5.1.1.3 Strstr

Synopsis

Function written in C to look for a substring in a string. Equivalent to the C function strcmp.

Input Ports

String1	string	substring to be located
String2	string	string in which to search for String1

Parameters

None

Output Ports

ret	boolean	true, substring found

Description

Strstr locates the first occurrence in String1 of the sequence of characters (excluding the terminating null character) in String2. Strstr returns true (1) if the string is found and false (0) if it is not.

Input Ports
String1
Substring to be located.

String2
String in which to locate String1.

Output Ports
ret
See Description above.

See also
-> $TOPDIR/inders3.project/src/Strstr.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Strstr.z
INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.1 String functions
5.1.1.4 Val

5.1.1.4 Val

Synopsis

Function written in C to convert a string to an integer. Equivalent to the C function atoi.

Input Ports

String	string	string to convert to integer

Parameters

None

Output Ports

Val	int	integer represented by String

Description

Val returns as an integer the value represented by String. The string is scanned up to the first non-digit. Leading white-space characters are ignored. If no integer can be found, zero is returned.

Input Ports
String
String to convert to integer

Output Ports
Val
See description above

See also
-> $TOPDIR/inders3.project/src/Val.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Val.z
INDERS documentation set
5 Miscellaneous
5.1 Functions
5.1.1 String functions
5.1.1.5 XValX

5.1.1.5 XVal

Synopsis

Function written in C to convert a string representing a hexadecimal number to an integer. Equivalent to the C function strtol with a base of 16.

Input Ports

String	string	hexadecimal string to convert to integer

Parameters

None

Output Ports

XVal	int	integer represented by String

Description

XVal returns as a integer the hexadecimal value represented by String. The string is scanned up to the first non-hexadecimal digit. Leading white-space characters are ignored. If no integer can be found, zero is returned.

Input Ports
String
Hexadecimal string to convert to integer

Output Ports
XVal
See description above

See also
-> $TOPDIR/inders3.project/src/XVal.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_XVal.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.1 String functions
5.1.1.6 get_man_names

5.1.1.6 get_man_names

Synopsis

INDERS3 function to find the names available in MAN_PATH and return them as two arrays.

Input Ports

trigger	int	start execution of function
MANpath	string	path to search for man pages

Parameters

None

Output Ports

V3_list	string[]	man pages for INDERS3 objects
Inders_list	string[]	man pages for INDERS1 codes

Description

get_man_names runs when trigger is changed. If MANpath is not specified, get_man_names uses the last entry in $MAN_PATH. get_man_names returns two arrays. One is the array of filenames that do not begin with "V3_Object" (assumed to be INDERS codes), without the trailing ".z". The other array is the filenames that begin with "V3_Object", but with "V3_Object_" stripped from filename and without the trailing ".z". These arrays are suitable for use as input to the UIoptionMenu.

Input Ports
trigger
Trigger to begin execution of get_man_names

MANpath
Directory to search for man pages. Default is last entry in $MAN_PATH.

Output Ports
V3_list
Array of man pages beginning "V3_Object".

Inders_list
Array of man pages not beginning "V3_Object".

See also

-> $TOPDIR/inders3.project/src/get_man_pages.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_get_man_pages.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.1 String functions
5.1.1.7 directory query

5.1.1.7 directory_query

Synopsis

INDERS3 function to determine whether the return from the file browser is a directory or a file.

Input Ports

Directory	string	string returned by a file or directory browser

Parameters

None

Output Ports

ret	boolean	one for directory, zero for file

Description

directory_query checks the last character of the Directory string. If the last character is a slash ("/"), ret is set to 1, otherwise a zero is returned. This is only needed on UNIX systems, since PC systems keep files and directories separate within the browers. In UNIX systems this string is not checked for validity when browsing. File browsers are allowed to return directories and directory browsers are allowed to return files. directory_query allows the user to check the validity within an applications.

Input Ports
Directory
Filename as selected by the user in UIfileSB.

Output Ports
ret
Set to one if Directory is a directory, and zero if Directory is a file.

See also

-> $TOPDIR/inders3.project/src/directory_query.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_directory_query.z

5.1.1.8 string_switch

Synopsis

general INDERS3 function to take two filenames and two testnames and output two parameters and a title.

Input Ports

filename_1	string	first filename to compare
filename_2	string	second filename to compare
testname_1	string	first test name to compare
testname_2	string	second test name to compare

Parameters

None

Output Ports

parameter_1	string	either filename_1 or testname_1
parameter_2	string	either filename_2 or testname_2
title	string	either filename_1 or testname_1

Description

string_switch compares the two filenames. If they are the same, then the parameters are set to the testnames and title is set to the filename. If the filenames are different, then the parameters are set to the filenames and title is set to the testname. If both filenames and testnames are different, nulls are returned.

Input Ports

filename_1
first filename to compare (required).

filename_2
second filename to compare (required).

testname_1
first test name to compare (required).

testname_2
second test name to compare (required).

Output Ports

parameter_1
either filename_1 or testname_1 (see Description above).

parameter_2
either filename_2 or testname_2 (see Description above).

title
either filename_1 or testname_1 (see Description above).

See also

-> $TOPDIR/inders3.project/src/string_switch.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_string_switch.z

5.1.2 General functions

5.1.2.1 arg_gen

Synopsis

INDERS3 function to concatenate several items with delimiters between them. The result is suitable as an argument to many external codes.

Input Ports

Input	string[]	array of entries to concatenate
Delimiter	string	delimiter to use between entries
FinalSpace	boolean	if set, include a final space
InitialDelimiter	boolean	if set, include an initial delimiter

Parameters

None

Output Ports

Arg	string	result of concatenation with delimiters

Description

arg_gen takes a string array and concatenates in the order the strings are attached to Input. A delimiter is placed between each entry and a final space and inital delimiter can be added.

Input Ports
Input

Array of strings to concatenate. The inputs can be other than string, as long as they can be automatically converted to string type by AVS. Thus int and float would be acceptable inputs.

Delimiter

String to use as a delimiter between Inputs. Can be a space (" "), slash ("/"), or any other character. The default is a comma (",").

FinalSpace

If set to one, a blank is appended to the end of the concatenated string. Default is one.

InitialDelimiter

If set to one, the first character of the concatenated string will be a delimiter. Useful for creating directory paths. Default is zero.

Output Ports
Arg

Result of string concatenation with delimiters added.

Example

```
arg_gen arg_gen {
    Input => { a, 1, b, 25};
};

will give
    arg_gen.Arg = "a,1,b,25 "

arg_gen arg_gen {
    Input => {"usr1","people","myname"};
    Delimiter = "/";
    FinalSpace = 0;
    InitialDelimiter = 1;
}

will give
    arg_gen.Arg = "/usr1/people/myname"
```

See also

-> $TOPDIR/inders3.project/src/arg_gen.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_arg_gen.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.2 General functions
5.1.2.2 filestatus

5.1.2.2 filestatus

Synopsis

INDERS3 function to determine whether the file browser is a file that can be correctly opened.

Input Ports

filename	string	string returned by a file or directory browser

Parameters

None

Output Ports

status	boolean	one if file can be successfully opened

Description

filestatus issues a fopen using filename. If the file is successfully opened, status is set to one. If the file cannot be opened, status is set to zero. The file is closed before filestatus exits.

Input Ports
filename
Filename as selected by the user in UIfileSB.

Output Ports
status
Set to one if file opens correctly.

See also

-> $TOPDIR/inders3.project/src/filestatus.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_filestatus.z

5.1.2.3 label_format

Synopsis

INDERS3 function to determine the best formatting for an array of floating point numbers.

Input Ports

values	float[]	values to determine format for

Parameters

None

Output Ports

format	string	format to be passed to str_format

Description

label_format looks through the values array to determine the range. At this time the set length of the format is 3 characters, so label_format determines the correct number of decimal points. This is used in labelling the legend.

Input Ports
values
An array of floating point values to be formatted.

Output Ports
format
Correct format for values array.

See also
-> $TOPDIR/inders3.project/src/label_format.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_label_format.z

5.1.2.4 make_array

Synopsis

INDERS3 function to make a smaller float array from a larger float array within a group.

Input Ports

array_group	&group	reference to a group containing an array
index	int	index in array_group to start sub-group

Parameters

None

Output Ports

out_array	float[]	output array

Description

make_array strips the input array out of the group. The output array consists of the elements of the input array starting at index and continuing to the end of the input array.

Input Ports
array_group
Reference to a group containing an array of numbers.

index
out_array's first element will be input_array[index].

Output Ports
out_array
The sub array starting at index.

See also
-> $TOPDIR/inders3.project/src/make_array.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_make_array.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.2 General_functions
5.1.2.5 make_cmd_file

5.1.2.5 make_cmd_file

Synopsis

INDERS3 function to determine the best formatting for an array of floating point numbers.

Input Ports

filename	filename	name of command file to create
cmd_line	string[]	file contents
trigger	int	cause file to be written

Parameters

None

Output Ports

done	boolean	set to one on successful completion

Description

make_cmd_file opens a file named filename for writing. If $HOME is included, it will be translated. The contents of cmd_line are concatenated with spaces between each element. A line feed is added before the final null in the complete string. The string is then written to filename. This is triggered by setting trigger to one. This is best accomplished using a UItoggle.

Input Ports
filename
Name of the command file to create. $HOME will be translated, if included.

cmd_line
Array with file contents. Generally, the lines to be written to $HOME/INDERS_queue.

trigger
When set to one, causes the file to actually be written. Can be connected to a UIbutton to give user control of file creation.

Output Ports
done
Inform other elements that file was successfully created.

See also
-> $TOPDIR/inders3.project/src/make_cmd_file.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_make_cmd_file.z

5.1.2.6 node_data_units

Synopsis

INDERS3 function to read node data units.

Input Ports

in	&group	input field

Parameters

None

Output Ports

Units	string[]	array of node units
Num_Units	int	size of the Units array

Description

node_data_units calls FLDget_node_data_ncomp to read the data units for a series of nodes from the field passed through "in". These are returned in the array Units.

Input Ports
in
Input field to read the node data units from.

Output Ports
Units
Values read from the field specifying the node data units

Num_Units
Array size of Units.

See also
-> $TOPDIR/inders3.project/src/node_data_units.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_node_data_units.z

5.1.2.7 str_sub_array

Synopsis

INDERS3 function to reduce a string array to a smaller number of elements.

Input Ports

str_array	&string[]	values to determine format for
index	int	index to copy to

Parameters

None

Output Ports

out_array	string[]	new array

Description

str_sub_array read the str_array and copies to out_array starting at the first element and continuing to "index".

Input Ports
str_array
Input string to be reduced.

index
Number of array elements to copy.

Output Ports
out_array
Array copied from str_array from element 1 to "index'

See also

-> $TOPDIR/inders3.project/src/str_sub_array.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_str_sub_array.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.2 General_functions
5.1.2.8 writecsv

5.1.2.8 writecsv

Synopsis

INDERS3 function to convert data into an MicroSoft Excel compatible comma seperated variable (CSV) format.

Input Ports

overwrite	int	if set, overwrite existing file
filename	string	CSV file to write
output	int	trigger to start file write
in	&field	input data field, or upper field (AWACS)
lower	&field	for AWACS, node data field
row_vals	float[]	for AWACS, row values
col_vals	float[]	for AWACS, column values
tolerance	float	for AWACS, tolerance value
JSF_toggle	int	if set, use JSF mesh, else use AWACS

Parameters

None

Output Ports

None

Description

Read data from "in" field and create a CSV field.

Input Ports
overwrite
If set, allow overwrite of existing file "filename".

filename
Name of the CSV file.

output
If set, begin writing file. This could be attached to a UIbutton to allow the user to start writing the file/

in
If JSF_toggle set, the data field to write to the CSV file. If JSF_toggle is not set, becomes the upper field.

lower
If JSF_toggle not set, the lower field.

row_vals
If JSF_toggle not set, values to be used for rows.

col_vals
If JSF_toggle not set, values to be used for columns.

tolerance
If JSF_toggle not set, tolerance.

JSF_toggle
If set, call JSF_Mesh, else call AWACS_CSV.

See also

-> $TOPDIR/inders3.project/src/writecsv.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_writecsv.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input functions

5.1.3 Input functions

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input functions
5.1.3.1 Waveform Info

5.1.3.1 Waveform_Info

Synopsis

INDERS3 structure to pass Waveform Information to Plot Macros and from Input Functions.

```
Waveform_Info {
    filename          string     filename to be read from
    Parameter_Name    string[]   parameters available to be plotted
    Parameters        float[]    parameter values
    Waveform          float[]    data values
    ordinate_name     string     axis name
    abscissa_name     string     axis name
    abscissa_start    float      beginning axis value
    max_col           int        number of columns in file
    max_row           int        number of files
    record_length     int        number of data points in waveform
}
```

Description

Group values are filled by various Input Functions. Values are then available to Plot Macros or to the user.

See also

-> $TOPDIR/inders3.project/v/functions.v
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_Waveform_Info.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input_functions
5.1.3.2 analytic_envelope

5.1.3.2 analytic_envelope

Synopsis

INDERS3 function to take a data array and find the envelope.

Input Ports

input	float[]	array to find envelope for
htfilter	float[]	Hilbert Tranform array
correction_factorfloat		correction factors

Parameters

None

Output Ports

output	float[]	envelope array

Description

analytic_envelope takes an input data array and a Hilbert Transform array and creates an envelope array.

Input Ports
input
An array of floating point values find an envelope for.

htfilter
An array of floating point values created by hilbert_transform.

correction_factor
Correction factor to apply to Hilbert Transformed data. Default is 1.0.

Output Ports
output
Envelope of input array.

See also

-> $TOPDIR/inders3.project/src/anal_env.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_analytic_envelope.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input_functions
5.1.3.3 bfs_header

5.1.3.3 bfs_header

Synopsis

INDERS3 function to take a data array and find the envelope.

Input Ports

filename	string	Blade/Fillet file to read

Parameters

None

Output Ports

x_inc	float	x increment
y_inc	float	y increment
Waveform_Info	group	See Waveform_Info

Description

bfs_header reads a Blade/Fillet file and returns x increment, y increment and also fills the Waveform_Info group. This provides information for read_gulp_array

Input Ports
filename
Blade/fillet filename

Output Ports
x_inc
Increment for x axis.

y_inc
Increment for y axis.

Waveform_Info
See Waveform_Info to see the values that must be filled in by bfs_header.

See also
-> $TOPDIR/inders3.project/src/bfs_header.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_bfs_header.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input_functions
5.1.3.4 drus_preface

5.1.3.4 drus_preface

Synopsis

INDERS3 function to read the header and coordinates of a DRUS file.

Input Ports

filename	string	DRUS file to read

Parameters

None

Output Ports

sample_interval	float[]	increment between samples
num_samps	int[]	number of samples per waveform
byte_loc	int[]	beginning location of waveform in file
coordinate {		
parameter	float[]	parameter values
}		
data_format	string	data format
Waveform_Info	group	See Waveform_Info

Description

drus_preface reads a DRUS file and fills in the above outputs. This information is used to read the specific waveform requested using rddrus.

Input Ports
filename
DRUS filename

Output Ports
sample_interval
Time increment between samples for each waveform.

num_samps
Number of samples per waveform.

byte_loc
Byte location in DRUS file of start of each waveform.

coordinate
Group containing parameter information for each waveform.

data_format
Data format of DRUS data within DRUS file.

Waveform_Info
See Waveform_Info to see the values that must be filled in by drus_preface.

See also
-> $TOPDIR/inders3.project/src/drus_preface.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_drus_preface.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input_functions
5.1.3.5 hilbert_tr

5.1.3.5 hilbert_tr

Synopsis

INDERS3 function to read the header and coordinates of a DRUS file.

Input Ports

halfwidth	int	halfwidth of Hilbert Filter

Parameters

None

Output Ports

HTFilter	float[]	Hilbert Filter

Description

hilbert_tr creates a Hilbert Tranform filter with halfwidth as specified. This filter is used by analytic_envelope.

Input Ports
halfwidth
specified halfwidth of Hilbert Filter. Must be a power of 2.

Output Ports
HTFilter
Hilbert Transform filter array. Array size is halfwidth/2.

See also
-> $TOPDIR/inders3.project/src/hilbert.c

-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_hilbert_tr.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input_functions
5.1.3.6 log_decode

5.1.3.6 log_decode

Synopsis

INDERS3 function to take data and remove any offset, then convert to log scale.

Input Ports

input	float[]	data array to process
bias	float	bias value of data
data_bits	int	number of bits in data

Parameters

None

Output Ports

output	float[]	log decoded array

Description

log_decode accepts data which can have an offset. The data is adjusted using "bias" to remove the offset. The adjusted data is then factored by 4/data_bits**2. Finally, the factored data is used to calculate the anti-log (10**factored_data). This array is returned in output.

Input Ports

input

Data to be transformed to log scale.

bias

Data offset. Will be subtracted from all data before processing. Default is 2047. Also determines sign of output data. If data element is less than bias, output will return a negative value for array element.

data_bits

Determines factor to be applied to offset-adjusted data. Factor is 4/data_bits**2. Default for data_bits is 12.

Output Ports

output

Log decoded array. Array size is equal to input array.

See also

-> $TOPDIR/inders3.project/src/log_decode.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_log_decode.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input_functions
5.1.3.7 mean_remove

5.1.3.7 mean_remove

Synopsis

INDERS3 function to adjust data to be centered around zero (subtract a "mean value" from the data set).

Input Ports

input	float[]	data to be adjusted
mean_start_index	int	location to start mean calculation
mean_end_index	int	location to end mean calculation

Parameters

None

Output Ports

output	float[]	mean-removed data

Description

mean_remove takes an average of the data elements of input between mean_start_index and mean_end_index, inclusive. This value is then subtracted from all elements of the input array to make the output array.

Input Ports

input

Data to be adjusted.

mean_start_index

First data element to be used in mean calculation.

mean_end_index

Final data element to be used in mean calculation.

Output Ports

output

Array of mean-removed data. Array size is equal to input array size.

See also

-> $TOPDIR/inders3.project/src/mean_remove.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_mean_remove.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input_functions
5.1.3.8 rddrus

5.1.3.8 rddrus

Synopsis

INDERS3 function to read a specific waveform from a DRUS file. drus_preface must have been executed first.

Input Ports

filename	string	DRUS file to read
Byte_Loc	int	start location of waveform
record_length	int	length of waveform
format	string	data format of waveform

Parameters

None

Output Ports

output	float[]	waveform data

Description

rddrus uses information from drus_preface to read one specific waveform from the DRUS file specified by filename.

Input Ports

filename

DRUS from which to read a waveform.

Byte_Loc

Starting location of specific waveform. This is retrieved from the byte_loc array filled by drus_preface. A UIslider can be used as an interface to the user to allow a specific waveform to be chosen.

record_length

Record length of a specific waveform. This is retrieved from the num_samps array filled by drus_preface. A UIslider can be used as an interface to the user to allow a specific waveform to be chosen. This should correspond to the same array element selected for Byte_Loc.

format

Data format of the waveform. This is retrieved from data_format as filled by drus_preface.

Output Ports

output

The waveform data array. Array size is record_length.

See also

-> $TOPDIR/inders3.project/src/rddrus.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_rddrus.z

INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input functions
5.1.3.9 rdsylk

5.1.3.9 rdsylk

Synopsis

INDERS3 function to read Thermocouple data from an Excel sylk file and convert to UCD format.

Input Ports

filename	string	Thermocouple file to read
minvalue	float	minimum value to include
maxvalue	float	maximum value to include
reduction_factor	float	amount to reduce "layers"
Output_Directory	string	directory in which to place the UCD file
largest_panel	float	largest panel size (lowest layer)

Parameters

None

Output Ports

number_of_fields	int	number of fields created
UCD_filename	string	name of file created

Description

rdsylk reads an Excel sylk file containing Thermocouple data. This is not a generic routine to read sylk files. The Thermocouple data is checked to see how long the thermocouple was above the minimum temperature. The number of points that exceed the maximum temperature is reported to the user.

The largest_panel and reduction_factor values are used to create visible panels varying in size from largest_panel at the lowest z, and reduced in size by reduction_factor for each next higher z. This allows the panels to be seen, even when overlapping in the viewer.

Input Ports

filename

Thermocouple file to convert to UCD.

minvalue

Lowest thermocouple value to use to calculate time above temperature.

maxvalue

If included, keep track of how many times this value was exceeded.

reduction_factor
Amount to reduce each layer when creating overlapping nodes. Default is 0.032

Output_Directory
Directory in which to write the UCD file.

largest_panel
Size of lowest and largest panel. Default is 0.75.

Output Ports
number_of_fields
Number of fields created in the UCD file. Zero indicates that the operation was unsuccessful.

UCD_filename
Name of the UCD file created.

See also
-> $TOPDIR/inders3.project/src/rdsylk_prolog.c
-> $TOPDIR/inders3.project/src/rdsylk_body.f
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_rdsylk.z
INDERS documentation set
Fusion
5 Miscellaneous
5.1 Functions
5.1.3 Input functions
5.1.3.10 read gulp array

5.1.3.10 read_gulp_array

Synopsis

INDERS3 function to read Blade/Fillet waveform data.

Input Ports

filename	string	Blade/Fillet data file to read
gulpOffset	int	location of waveform to be read
gulpLength	int	length of waveform to be read
byteOffset	int	number of bytes to skip at start of file

Parameters

None

Output Ports

output	int[]	Blade/Fillet waveform data

Description

read_gulp_array reads a waveform from a blade/fillet file. Blade/fillet data is stored in "gulps" (3 bytes). These "gulps" are in little endian format. Conversion is made to big endian where needed (UNIX machines).

Input Ports
filename
Blade/fillet file from which to read waveform.

gulpOffset
Location (in gulps) of waveform to be read. Calculated based on gulpLength and user requested waveform.

gulpLength
Length (in gulps) of waveform to be read. This value is supplied by bfs_header in the Waveform_Info variable record_length.

byteOffset
Number of bytes to discard at the start of the file. Default is 512.

Output Ports
output

Blade/fillet waveform data.

See also

-> $TOPDIR/inders3.project/src/read_gulp_data.c
-> $TOPDIR/inders/runtime/catman/a_man/cat1/V3_Object_read_gulp_data.z

Fusion
5 Miscellaneous
5.2 MultiFields

5.2 MultiFields

INDERS documentation set
Fusion
5 Miscellaneous
5.2 MultiFields
5.2.2 Generic MultiField

5.2.2 Generic MulitField

Synopsis

General INDERS3 visualizable data object. Used for transforming data coordinates.

Input Ports

Field	Field	unstructured field containing data
visible	int	Generic_MultiField GUI visibile when set
parent	UIconnection	connect to UI display device

Parameters

Select	UIoptionMenu	display Units or Equations
X1 to X3	UIlabel	Units or Coordinates from file
X4 to X6	UIfield	Units/Equations for 1st transform
X7 to X9	UIfield	Units/Equations for 2nd transform
X10 to X12	UIfield	Units/Equations for 3rd transform

Output Ports

Tfield2	Field	transformed field output

Description

Generic_MultiField takes an unstructured field (AVS/Express field type) and transforms it into another unstructured field containing the same data as the input field but with different coordinates which are defined by user programmable transform equations. There are three sets of transformations provided (for a total of 12 coordinates - the first set of 3 being the original coordinates and the fourth and final set of 3 being the visualization coordinates).

The result is a visualizable rendering of the triply transformed field output displaying a user selectable component of the field data with respect to the fourth set of coordinates. The fields created by the intermediate transforms are available within the object using the network editor.

Generic_MultiField also contains coordinates unit definitions and a MultiFieldHeader which are available to user applications for decision making.

Input Ports

Field
Typically connected to an unstructured field (AVS/Express field type). For instance, the output of an AVS/Express Read_UCD object.

visible
Must be set to "1" in order to see the Generic_MultiField GUI display.

parent
Must be connected to a UI display device, such as a UImodPanel where the Generic_MultiField is to be displayed.

Parameters
Select
UIoptionMenu. Choose to display either Units or Defining Equations.

X1 to X3
UIlabel. Display coordinate names or units for first three coordinates (eg: X, Y, Z). These values are read from the file. For information only.

X4 to X6
UIfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is no transformation.

X7 to X9
UIfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is no transformation.

X10 to X12
UIfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is no transformation.

Output Ports
Tfield2
A visualizable rendering of the triply transformed field output displaying a user selectable component of the field data with respect to the fourth set of coordinates.

INDERS documentation set
Fusion
5 Miscellaneous
5.2 MultiFields
5.2.3 JSF MultiField

5.2.3 JSF MulitField

Synopsis
JSF specific INDERS3 visualizable data object. Used for transforming data coordinates for mapping onto JSF Spar.

Input Ports

Field	Field	unstructured field containing data
visible	int	JSF_MultiField GUI visibile when set
parent	UIconnection	connect to UI display device

Parameters

Select	UIoptionMenu	display Units or Equations
X1 to X3	UIlabel	Units or Coordinates from file
X4 to X6	UIfield	Units/Equations for 1st transform
X7 to X9	UIfield	Units/Equations for 2nd transform
X10 to X12	UIfield	Units/Equations for 3rd transform

Output Ports

Tfield2	Field	transformed field output

Description

JSF_MultiField is built from the Generic_MultiField. It has JSF Spar specific defaults built into the coordinate tranformation equations. These equations are also user modifiable.

The result is a visualizable rendering of the field that will map onto a JSF Spar object.

JSF_MultiField also contains coordinates unit definitions and a MultiFieldHeader which are available to user applications for decision making.

Input Ports
Field
Typically connected to an unstructured field (AVS/Express field type). For instance, the output of an AVS/Express Read_UCD object.

visible
Must be set to "1" in order to see the JSF_MultiField GUI display.

parent
Must be connected to a UI display device, such as a UImodPanel where the JSF_MultiField is to be displayed.

Parameters
Select
UIoptionMenu. Choose to display either Units or Defining Equations.

X1 to X3
UIlabel. Display coordinate names or units for first three coordinates (eg: X, Y, Z). These values are read from the file. For information only.

X4 to X6
UIfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is X4 = X1+0.5, X5 = X2, and X6 = X3 + 0.25.

X7 to X9
UIfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is no transformation.

X10 to X12
UIfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is no transformation.

Output Ports
Tfield2
A visualizable rendering of the triply transformed field output displaying a user selectable component of the field data with respect to the fourth set of coordinates.

INDERS documentation set
Fusion
5 Miscellaneous
5.2 MultiFields
5.2.4 AWACS tm MultiField

5.2.4 AWACS tm MulitField

Synopsis
AWACS specific INDERS3 visualizable data object. Used for transforming data coordinates for mapping onto an AWACS Radome.

Input Ports

Field	Field	unstructured field containing data
visible	int	AWACS_MultiField GUI visibile when set
parent	UIconnection	connect to UI display device

Parameters

Select	UIoptionMenu	display Units or Equations
td	UIlabel	circumferential position from file (degrees)
m	UIlabel	meridional position from file
b	UIlabel	blank (ignored)
tr	UIfield	circumferential position (radians)
z	UIfield	Units/Equations for z
r	UIfield	Units/Equations for radius
x	UIfield	Units/Equations for x

y	UIfield	Units/Equations for y
unused	UIfield	ignored
xviewer	UIfield	Units/Equations for xviewer
yviewer	UIfield	Units/Equations for yviewer
zviewer	UIfield	Units/Equations for zviewer

Output Ports

Tfield2	Field	transformed field output

Description

AWACS_tm_MultiField is built from the Generic_MultiField. It has AWACS_Radome specific defaults built into the coordinate tranformation equations. These equations are also user modifiable.

The result is a visualizable rendering of the field that will map onto a AWACS_Radome object.

AWACS_tm_MultiField also contains coordinates unit definitions and a MultiFieldHeader which are available to user applications for decision making.

Input Ports

Field

Typically connected to an unstructured field (AVS/Express field type). For instance, the output of an AVS/Express Read_UCD object.

visible

Must be set to "1" in order to see the AWACS_tm_MultiField GUI display.

parent

Must be connected to a UI display device, such as a UImodPanel where the AWACS_tm_MultiField is to be displayed.

Parameters

Select

UIoptionMenu. Choose to display either Units or Defining Equations.

td

UIlabel. Display coordinate name or units for circumferential position from file (degrees). These values are the first coordinate read from the file. For information only.

m

UIlabel. Display coordinate name or units for meridional position from file. These values are the second coordinate read from the file. For information only.

b

UIlabel. Display coordinate name or units for third coordinate read from the file. Not used in this application.

tr

UIfield. Display transformation equations or units for circumferential position (radians). Default is tr = td * RADIANS_PER_DEGREE (defined in dictionary).

z

UIfield. Display transformation equations or units for z position. Default is the transform zra.

r

UIfield. Display transformation equations or units for radial position. Default is r = sqrt (180**2 - 5 * z**2).

x

UIfield. Display transformation equations or units for x position. Default is x = r * cos(tr).

y

UIfield. Display transformation equations or units for y position. Default is y = r * sin(tr).

unused

UIfield. Blank (ignored)

xviewer

UIfield. Display transformation equations or units for x viewer position. Default is xviewer = x.

yviewer

UIfield. Display transformation equations or units for x viewer position. Default is yviewer = y.

zviewer

UIfield. Display transformation equations or units for x viewer position. Default is zviewer = z.

Output Ports

Tfield2

A visualizable rendering of the triply transformed field output displaying a user selectable component of the field data with respect to the fourth set of coordinates.

5.2.5 AWACS e MulitField

Synopsis

AWACS specific INDERS3 visualizable data object. Used for transforming data coordinates for mapping onto an AWACS Radome.

Input Ports

Field	Field	unstructured field containing data
visible	int	AWACS_MultiField GUI visibile when set
parent	UIconnection	connect to UI display device

Parameters

Select	UIoptionMenu	display Units or Equations
m	UIlabel	meridional position from file
td	UIlabel	circumferential position from file (radians)
b	UIlabel	blank (ignored)
tr	UIfield	circumferential position (radians)
z	UIfield	Units/Equations for z
r	UIfield	Units/Equations for radius
x	UIfield	Units/Equations for x
y	UIfield	Units/Equations for y
unused	UIfield	ignored
xviewer	UIfield	Units/Equations for xviewer
yviewer	UIfield	Units/Equations for yviewer
zviewer	UIfield	Units/Equations for zviewer

Output Ports

Tfield2	Field	transformed field output

Description

AWACS_e_MultiField is built from the Generic_MultiField. It has AWACS_Radome specific defaults built into the coordinate tranformation equations. These equations are also user modifiable.

The result is a visualizable rendering of the field that will map onto a AWACS_Radome object.

AWACS_e_MultiField also contains coordinates unit definitions and a MultiFieldHeader which are available to user applications for decision making.

Input Ports

Field

Typically connected to an unstructured field (AVS/Express field type). For instance, the output of an AVS/Express Read_UCD object.

visible

Must be set to "1" in order to see the AWACS_e_MultiField GUI display.

parent

Must be connected to a UI display device, such as a UImodPanel where the AWACS_e_MultiField is to be displayed.

Parameters

Select

UIoptionMenu. Choose to display either Units or Defining Equations.

td
Ullabel. Display coordinate name or units for circumferential position from file (radians). These values are the second coordinate read from the file. For information only.

m
Ullabel. Display coordinate name or units for meridional position from file. These values are the first coordinate read from the file. For information only.

b
Ullabel. Display coordinate name or units for third coordinate read from the file. Not used in this applications.

tr
Ulfield. Display transformation equations or units for circumferential position (radians). Default is tr = td * RADIANS_PER_DEGREE (defined in dictionary).

z
Ulfield. Display transformation equations or units for z position. Default is the transform z = td. Not used in this application.

r
Ulfield. Display transformation equations or units for radial position. Default is r = td. Not used in this application.

x
Ulfield. Display transformation equations or units for x position. Default is x = td. Not used in this application.

y
Ulfield. Display transformation equations or units for y position. Default is y = td. Not used in this application.

unused
Ulfield. Blank (ignored)

xviewer
Ulfield. Display transformation equations or units for x viewer position. Default is xviewer = 180 * cos(tr).

yviewer
Ulfield. Display transformation equations or units for x viewer position. Default is yviewer = m + 999.

zviewer
Ulfield. Display transformation equations or units for x viewer position. Default is zviewer = 180 * sin(tr).

Output Ports
Tfield2
A visualizable rendering of the triply transformed field output displaying a user selectable component of the field data with respect to the fourth set of coordinates.

INDERS documentation set
Fusion
5 Miscellaneous
5.2 MultiFields
5.2.6 AUSS 90in MultiField

5.2.6 AUSS 90in MultiField

Synopsis
B1 specific INDERS3 visualizable data object. Used for transforming data coordinates for mapping onto B1_90inDoor.

Input Ports

Field	Field	unstructured field containing data
visible	int	AUSS_90in_MultiField GUI visibile when set
parent	Ulconnection	connect to UI display device

Parameters

Select	UloptionMenu	display Units or Equations
X1 to X3	Ullabel	Units or Coordinates from file
X4 to X6	Ulfield	Units/Equations for 1st transform

X7 to X9	UIfield	Units/Equations for 2nd transform
X10 to X12	UIfield	Units/Equations for 3rd transform

Output Ports

Tfield2	Field	transformed field output

Description

AUSS_90in_MultiField is built from the Generic MultiField. It has AUSS 90in Data specific defaults built into the coordinate tranformation equations. These equations are also user modifiable.

The result is a visualizable rendering of the field that will map onto a B1 90inDoor object.

AUSS_90in_MultiField also contains coordinates unit definitions and a MultiFieldHeader which are available to user applications for decision making.

Input Ports

Field

Typically connected to an unstructured field (AVS/Express field type). For instance, the output of an AVS/Express Read_UCD object.

visible

Must be set to "1" in order to see the AUSS_90in_MultiField GUI display.

parent

Must be connected to a UI display device, such as a UImodPanel where the AUSS_90in_MultiField is to be displayed.

Parameters

Select

UIoptionMenu. Choose to display either Units or Defining Equations.

X1 to X3

UIlabel. Display coordinate names or units for first three coordinates (eg: X, Y, Z). These values are read from the file. For information only.

X4 to X6

UIfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is X4 = X1+ 685, X5 = -X2 + 45, and X6 = X3.

X7 to X9

UIfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is no transformation.

X10 to X12

UIfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is no transformation.

Output Ports

Tfield2

A visualizable rendering of the triply transformed field output displaying a user selectable component of the field data with respect to the fourth set of coordinates.

INDERS documentation set
Fusion
5 Miscellaneous
5.2 MultiFields
5.2.7 AUSS 180in MultiField

5.2.6 AUSS 180in MultiField

Synopsis

B1 specific INDERS3 visualizable data object. Used for transforming data coordinates for mapping onto B1 180inDoor.

Input Ports

Field	Field	unstructured field containing data
visible	int	AUSS_180in_MultiField GUI visibile when set

parent	Ulconnection	connect to UI display device

Parameters

Select	UloptionMenu	display Units or Equations
X1 to X3	Ullabel	Units or Coordinates from file
X4 to X6	Ulfield	Units/Equations for 1st transform
X7 to X9	Ulfield	Units/Equations for 2nd transform
X10 to X12	Ulfield	Units/Equations for 3rd transform

Output Ports

Tfield2	Field	transformed field output

Description

AUSS_180in_MultiField is built from the Generic_MultiField. It has AUSS_180in_Data specific defaults built into the coordinate tranformation equations. These equations are also user modifiable.

The result is a visualizable rendering of the field that will map onto a B1_180inDoor object.

AUSS_180in_MultiField also contains coordinates unit definitions and a MultiFieldHeader which are available to user applications for decision making.

Input Ports

Field

Typically connected to an unstructured field (AVS/Express field type). For instance, the output of an AVS/Express Read_UCD object.

visible

Must be set to "1" in order to see the AUSS_180in_MultiField GUI display.

parent

Must be connected to a UI display device, such as a UlmodPanel where the AUSS_180in_MultiField is to be displayed.

Parameters

Select

UloptionMenu. Choose to display either Units or Defining Equations.

X1 to X3

Ullabel. Display coordinate names or units for first three coordinates (eg: X, Y, Z). These values are read from the file. For information only.

X4 to X6

Ulfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is X4 = X1+ 0.5, X5 = -X2 + 45, and X6 = X3.

X7 to X9

Ulfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is no transformation.

X10 to X12

Ulfield. Display transformation equations or units for the first of 3 sets of transformations on X1 to X3 (the first three coordinates). This allows mapping of the data to complex part surfaces. Default is no transformation.

Output Ports

Tfield2

A visualizable rendering of the triply transformed field output displaying a user selectable component of the field data with respect to the fourth set of coordinates.